Algorithms seminar, 1994-1995

Bruno Salvy

INRIA
INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET AUTOMATIQUE

Algorithms seminar, 1994-1995

Bruno SALVY, éditeur scientifique

N ° 2669
Octobre 1995

PROGRAMME 2

ISSN 0249-6399

ALGORITHMS SEMINAR,

1994-1995

Bruno Salvy

(Editor)

Abstract

These seminar notes represent the proceedings of a seminar devoted to the analysis of algorithms and related topics. The subjects covered include combinatorics, symbolic computation, asymptotic analysis, average-case analysis of algorithms and data structures, and computational number theory.

SÉMINAIRE ALGORITHMES,

1994-1995

Abstract

Ces notes de séminaires représentent les actes, pour la plupart en anglais, d'un séminaire consacré à l'analyse d'algorithmes et aux domaines connexes. Les thèmes abordés comprennent : combinatoire, calcul formel, analyse asymptotique, analyse en moyenne d'algorithmes et de structures de données, ainsi que de la théorie algorithmique des nombres.

ALGORITHMS SEMINAR
1994–1995

Bruno Salvy[1]
(Editor)

Abstract

These seminar notes represent the proceedings of a seminar devoted to the analysis of algorithms and related topics. The subjects covered include combinatorics, symbolic computation, asymptotic analysis, average-case analysis of algorithms and data structures, and computational number theory.

This is the fourth of our series of seminar proceedings. The previous ones have appeared as INRIA Research Reports numbers 1779, 2130 and 2381. The content of these proceedings consists of summaries of the talks, usually written by a reporter from the audience.

The primary goal of this seminar is to cover the major methods of the average-case analysis of algorithms and data structures. Neighbouring topics of study are combinatorics, symbolic computation and asymptotic analysis.

Several articles deal with combinatorial objects—their description or their random generation—that are useful for simulations and empirical studies.

Computer algebra plays an increasingly important rôle in this area. It provides a collection of tools that permit to attack complex models of combinatorics and of the analysis of algorithms; at the same time, it inspires the quest for developing ever more systematic solutions to the analysis of well characterized classes of problems. In this vein, the notes contain several recent developments regarding the automatic manipulation of differential and recurrence equations.

Asymptotic methods have been covered extensively in the previous volumes of this seminar. These proceedings include approaches to divide-and-conquer recurrences and to some probabilistic functions with applications to several algorithms.

The 31 articles included in this book represent snapshots of current research in these areas. A tentative organization of their contents is given below.

PART I. COMBINATORICS

In addition to its own traditions rooted in mathematics, the study of *combinatorial models* arises naturally in the process of analyzing algorithms that often involve classical combinatorial structures like strings, trees, graphs, permutations.

In [**1**], an algorithm for random generation of sets of (i.e. without repetition) of various combinatorial objects is described. Aspects of the study of RNA are the subject of [**2**]. Automatic sequences and their complexity are discussed in [**3**]. Some double sequences and their asymptotic properties are studied in [**4**]. Finally, [**5**] gives an arithmetic interpretation to simple operations on binary trees.

[1]This work was supported in part by the ESPRIT III Basic Research Action Programme of the E.C. under contract ALCOM II (#7141).

[1] Uniform Random Generation for the Powerset Construction. *Paul Zimmermann*
[2] An Efficient Parser Well Suited to RNA Folding. *Fabrice Lefebvre*
[3] Pascal's Triangle, Automata, and Music. *Jean-Paul Allouche*
[4] Riordan Arrays and their Applications. *Donatella Merlini*
[5] Structured Numbers. *Vincent Blondel*

PART II. SYMBOLIC COMPUTATION

Most exactly solvable models of combinatorics and analysis of algorithms rest on a suitable algebra of *generating functions*. Once this has been recognized, an important goal is to find decision procedures for classes of generating functions. Computer algebra systems provide a way of testing and implementing the methods, and the problem of optimizing the corresponding procedures often represents a non-trivial problem of symbolic computation.

An important class of generating functions is formed by solutions of linear differential equations. Alternatively, the coefficients of these generating functions can be given by a linear recurrence. Any kind of solution to these equations can be used to help the analysis. Polynomial solutions of very general linear equations can be found algorithmically [7]. Divergent series often occur as solutions to linear differential equations, but several algorithms make it possible to deal with them [8]. A general framework for the manipulation of linear operators and proof of combinatorial identities is the subject of [9]. In [10], a remark about the use of such techniques for q-identities yields short proofs of some of the Rogers-Ramanujan identities. Series defined by systems of non-linear differential equations can be manipulated effectively, and in particular their equivalence problem is solvable [11]. A framework for asymptotic expansions by computer algebra is given in [12]. Other computer algebra topics are the determination of the sign of determinants with single-precision arithmetic [6], fast computation on matrices over a finite field [13], factorization over finite fields [14], efficient computations with algebraic curves [15], and integration of hyperelliptic functions [16].

[6] Evaluating Signs of Determinants. *Jean-Daniel Boissonnat*
[7] Polynomial Solutions of Linear Operator Equations. *Marko Petkovšek*
[8] Symbolic and Numerical Manipulations of Divergent Power Series. *Jean Thomann*
[9] Holonomic Systems and Automatic Proofs of Identities. *Frédéric Chyzak*
[10] Short and Easy Computer Proofs of Partition and q-Identities. *Peter Paule*
[11] Effective Identity Testing in Extensions of Differential Fields. *Ariane Péladan-Germa*
[12] Automatic Asymptotics. *Joris van der Hoeven*
[13] Normal Bases and Canonical Rational Form (Over Finite Fields). *Daniel Augot*
[14] Factoring Polynomials Over Finite Fields. *Daniel Panario*
[15] The Integral Basis of an Algebraic Function Field. *Mark van Hoeij*
[16] Symbolic Computation of Hyperelliptic Integrals. *Laurent Bertrand*

PART III. ASYMPTOTIC ANALYSIS

Asymptotic analysis is an essential ingredient in the interpretation of quantitative results supplied by the resolution of combinatorial models.

An important class of problems involves recovering the asymptotic form of the coefficients of a function from asymptotic properties of the function itself. This approach is taken in [17] to analyze some divide-and-conquer recurrences and in [19] to study statistics related to some probabilistic algorithms. The asymptotic behaviour of solutions to some non-linear differential equations is described in [18]. In [20], structural limitations of some classes of asymptotic expansions are pointed out.

PART IV. ANALYSIS OF ALGORITHMS AND DATA STRUCTURES

This part deals with the analysis of algorithms and data structures.

While functional analysis is the main tool of [21] in the analysis of the Gauss reduction algorithm, singularity analysis is applied in [22] to algorithms that generalize symbolic differentiation, and properties of the Mellin transform are exploited in [24] to analyze communication protocols. An algorithm for random generation is analyzed in [23]. Another algorithm for solving a problem that arises in automatic differentiation is given in [26], and the last two notes describe problems related to computational genetics.

PART V. MISCELLANY

This part contains an introduction to complex multiplication [29], an introduction to simulated annealing [30] and an algebraic framework for computation with multivariate rational functions [31].

Acknowledgements. The lectures summarized here emanate from a seminar attended by a community of researchers in the analysis of algorithms, in the Algorithms Project at INRIA (Ph. Flajolet, F. Morain and B. Salvy are the organizers) and in the greater Paris area—especially École Polytechnique (J.-M. Steyaert), University of Paris Sud at Orsay (D. Gouyou-Beauchamps) and LITP (M. Soria).

The editor expresses his gratitude to the various persons who supported actively this joint enterprise, most notably: Philippe Dumas for his careful rereading, Eithne Murray for straightening some English texts, and Frédéric Chyzak. Thanks are due also to the speakers and to the authors of summaries. Many of them have come from far away to attend one seminar and nicely accepted to write the summary.

We are also greatly indebted to Virginie Collette for making all the organization work smoothly.

The Editor
B. SALVY

Part 1

Combinatorics

Uniform Random Generation for the Powerset Construction

Paul Zimmermann
Inria Lorraine

December 12, 1994

[summary by Eithne Murray]

Abstract

An algorithm for the uniform random generation of the powerset construction is presented. Given a combinatorial class I, together with a counting procedure and an unranking procedure (or simply a random generation procedure) for I, this algorithm provides counting and unranking (or random generation) procedures for $P = \texttt{powerset}(I)$. For most combinatorial structures, each random powerset of size n is produced in $\mathcal{O}(n \log n)$ arithmetic operations in the worst case, after $\mathcal{O}(n^2)$ coefficients have been computed. This work is an extension of the algorithms developed in [1, 2], that have been implemented in the Gaïa (now combstruct) Maple package [4].

1. Introduction

Given a combinatorial class I of unlabelled objects such that for each integer n the number $I[n]$ of objects of size n is finite (and, for convenience, $I[0] = 0$), the problem is to generate uniformly at random an object of size n from $\texttt{powerset}(I)$, where $\texttt{powerset}(I)$ means the class of sets without repetition made from objects in I.

Associate with each combinatorial class I two procedures — a counting procedure `countI`, such that $\texttt{countI}(n)$ gives the number $I(n)$ of objects of size n, and an unranking procedure `unrankI` which implements a bijection between $[0, I(n) - 1]$ and the set of objects of size n from I. Thus $\texttt{unrankI}(n, k)$ returns the object of size n whose *rank* is k.

Given these two procedures, the algorithm constructs similar functions `countP` and `unrankP` to count and generate at random the objects from $P = \texttt{powerset}(I)$.

The article [5] this seminar is based on is available from http://www.loria.fr/~zimmerma/gaia. It contains implementation details, proofs, examples and experimental results.

2. The Counting Problem

The generating functions $P(z)$ and $I(z)$ satisfy this identity due to Pólya: $P(z) = \exp(I(z) - \frac{1}{2}I(z^2) + \frac{1}{3}I(z^3) - \frac{1}{4}I(z^4) + \cdots)$. Using this identity and the operator $\Theta = z\,d/dz$ (see [1]), it is easy to obtain equations to compute the coefficients $P[k] = [z^k]P(z)$ for $k = 1, ..., n$ in $\mathcal{O}(n^2)$ operations.

3. An Unranking Algorithm

To generate a random object of size n from a powerset, consider the problem in two steps. First, generate the *shape* the generated object will have, that is, how many objects of each size will be present in the set. Second, for each size of object in the set, generate the objects of that size.

3.1. Shape Generation. To generate a powerset of size n, the algorithm chooses non-negative integers $(i_1, ..., i_k)$ such that $n = i_1 + 2i_2 + \cdots + ki_k$. It must ensure that the shape is generated with a probability based on the number of powersets having the shape $(i_1, ..., i_k)$. This probability depends only on the numbers $I[\ell]$, for ℓ between 1 and n. Thus, only the procedure `countI` is needed to solve the shape generation problem. This algorithm is based on the decomposition $P_{\ell,m} = \texttt{Prod}(P_{\ell,2m}, P_{\ell+m,2m})$, where $P_{\ell,m}$ is the set of objects of size $\ell + \lambda m$, where λ is a non-negative integer, m is a power of 2, $1 \leq \ell \leq m$, and `Prod` denotes the cartesian product. It involves calculating $\mathcal{O}(n^2)$ sizes of $P_{\ell,m}$.

3.2. Equal Size Generation. The equal size generation problem is equivalent to the problem of selecting k distinct elements $a_1, ..., a_k$ from $1, 2, ..., n$ (called "selection sampling" by Knuth [3]). There appears to be no known algorithm to do unranking for this problem in $\mathcal{O}(k)$ time and space. P. Zimmermann proposes an unranking algorithm for the selection sampling problem that has $\mathcal{O}(k \log n)$ worst case complexity.

A method of unranking powersets is then obtained by using this selection sampling algorithm. By replacing the unranking procedure for I in the unranking algorithm by a random generation procedure, a random generation procedure for P is also formed. The complexity analysis for the unranking algorithm depends on a condition of *standard growth* on the combinatorial class, while the analysis for the random generation algorithm holds for all combinatorial classes. Most combinatorial classes satisfy this condition, but otherwise, such as for a recursive structure where I depends on P, there is little information about the algorithm's efficiency.

DEFINITION 1. A combinatorial class I is of *standard growth* if their exists a constant A such that the number $I[n]$ of structures of size n satisfies $I[n] \leq n^{An}$ for n sufficiently large.

THEOREM 1. *If I is any combinatorial class, and* `randomI`(n) *has average cost* $\mathcal{O}(n \log n)$*, then* `randomP`(n, k) *also has average cost* $\mathcal{O}(n \log n)$*. If I is a combinatorial class of standard growth and* `unrankI`(n, k) *has worst case* $\mathcal{O}(n \log n)$*, then* `unrankP`(n, k) *also has worst case cost* $\mathcal{O}(n \log n)$*.*

4. Conclusion and Open Questions

Given any combinatorial class I, a counting procedure for I, and a procedure for unranking (random generation) for I, there is an algorithm to do unranking and random generation for $P = \texttt{powerset}(I)$. Some questions remain. Is there an $\mathcal{O}(k)$ algorithm for unranking k-samples in an n-set? And what is the worst case complexity of this algorithm when I has a recursive specification? Also, the pre-processing time of this algorithm is rather high ($\mathcal{O}(n^2)$ operations and about $\mathcal{O}(n^4)$ time). This pre-processing cost should be reduced.

Bibliography

[1] Flajolet (Philippe), Zimmerman (Paul), and Van Cutsem (Bernard). – A calculus for the random generation of labelled combinatorial structures. *Theoretical Computer Science*, vol. 132, n° 1-2, 1994, pp. 1–35.

[2] Flajolet (Philippe), Zimmermann (Paul), and Van Cutsem (Bernard). – A calculus of random generation: Unlabelled structures. In preparation.

[3] Knuth (Donald E.). – *The Art of Computer Programming.* – Addison-Wesley, 1981, 2nd edition, vol. 2: Seminumerical Algorithms.

[4] Zimmermann (Paul). – Gaïa: A package for the random generation of combinatorial structures. *MapleTech*, vol. 1, n° 1, 1994, pp. 38–46.

[5] Zimmermann (Paul). – Uniform random generation for the powerset construction. In Leclerc (B.) and Thibon (J. Y.) (editors), *Formal power series and algebraic combinatorics.* pp. 589–600. – Université de Marne-la-Vallée, 1995. Proceedings SFCA'95.

An Efficient Parser Well Suited to RNA Folding

Fabrice Lefebvre
LIX, École polytechnique

26 Juin 1995

[summary by Fabrice Lefebvre]

1. Introduction

In RNA, interactions between nucleotides form base pairs and, seen at a higher level, characteristic secondary structure motifs such as helices, loops and bulges. These motifs are of great interest to biologists. Though the secondary structure of RNA is much simpler than its tertiary structure, it remains difficult to compute because the number of secondary structures of an RNA of n bases grows exponentially with n [9]. Several methods have been established for folding RNAs, that is predicting RNA secondary structure. The first method is phylogenetic analysis of homologous RNA molecules. It relies on conservation of structural features during evolution. Some people are trying to apply a grammar formalism to this method [3]. The second method uses a simplified thermodynamic model of RNA secondary structure to find the structure with the lowest free energy. The third method has been recently introduced by Haussler *et al.* [6] and it relies on *stochastic context-free grammars* (SCFGs) to model common secondary structures of a given family of RNAs. Our parser has been designed to express easily the latest two methods.

2. Folding and S-attribute grammars

It is well known that secondary structures without pseudo-knots of an RNA may be seen as derivation trees of this RNA for a suitably defined context-free grammar (CFG) [7]. We might for instance use the following grammar with terminals A, C, G, U:

$$E \to \epsilon \mid AE \mid CE \mid GE \mid UE \mid AEUE \mid UEAE \mid GECE \mid GEUE \mid CEGE \mid UEGE$$

We shall use the following classic definition of context-free grammars (CFGs).

Definition 1. A CFG $G = (T, N, P, S)$ consists of finite sets of terminals T, nonterminals N, productions (rewriting rules) P and of a start symbol $S \in N$. Let $V = N \cup T$ denote the vocabulary of the grammar. Each production in P has the form $A \to \alpha$, where $A \in N$ and $\alpha \in V^*$. A is the left-hand side of the production and α its right-hand side.

In our work, we assumed that the grammar is proper (no useless rules or symbols, non-circular, epsilon-free). Grammars whose derivation trees describe secondary structures will always be ambiguous because a given RNA always has many different secondary structures.

CFGs allow us to give a synthetic description of a set of secondary structures, but they do not allow us to choose one structure among this set. S-attribute CFGs (S-ACFGs) [4] are an extension of CFGs allowing the assignment of a value (called attribute) to every vertex of a derivation tree. With attributes, we may now select derivation trees with a simple criterion. If the attribute of a

vertex is an energy or a probability, the criterion may be the selection of the derivation tree with the lowest energy or the highest probability at the root. But attributes are not restricted to simple real values and may be more complex. Our context of utilization of S-ACFGs has led us to the following definition for those grammars.

DEFINITION 2. An S-ACFG is denoted by $G = (T, N, P, S, \mathcal{A}, S_{\mathcal{A}}, F_P)$. This is an extension of the proper CFG $G = (T, N, P, S)$, where an attribute $x \in \mathcal{A}$ is attached to each symbol $X \in V$, and a string of attributes $\lambda \in \mathcal{A}^*$ to each string $\alpha \in V^*$. $S_{\mathcal{A}}$ is a function from T to $\mathcal{A}$ assigning attributes to terminals. F_P is a set of functions from $\mathcal{A}^*$ to $\mathcal{A}$. A function $f_{A\to\alpha}$ is in F_P iff $A \to \alpha$ is in P.

The attribute λ of a string α is the concatenation of the attributes of the symbols in α. When a function $f_{A\to\alpha}$ is applied to the attribute λ of a string α derived from A, it returns the attribute x of A. Thus, functions of F_P are responsible for the computation, in a bottom-up way, of the attributes of nonterminals A in derivations $\alpha A\beta \to^* u$, where u must belong to T^* in order that the attribute of A may be computable.

3. Two known parsing algorithms

With our application of parsing to RNAs, we will have to find one derivation tree among a potentially exponential number of derivation trees. Hence tabular algorithms are a good way to deal with this parse forest since they output a compacted representation of the parse forest in polynomial time $O(n^3)$ and space $O(n^2)$.

The simplest algorithm is the one of Cocke-Younger-Kasami [1]:
Let G be a proper CFG in Chomsky normal form. The algorithm builds a table $(T_j)_{1\leq j\leq n}$ such that T_j contains the item $[A, i]$ iff $A \to a_{i+1} \dots a_j$.

For j between 1 and n, perform the following steps

(1) Add $[A, j-1]$ to T_j if $A \to a_j$;
(2) Add $[A, i]$ to T_j if $\exists k < j$ such that $[B, i] \in T_k$ and $[C, k] \in T_j$ and $A \to BC$;
(3) Repeat the previous step while there remains items to be added to T_j.

The string $a_1 \dots a_n$ belongs to $L(G)$ iff $[S, 0] \in T_n$.

CYK's algorithm needs grammars in Chomsky normal form, and it does not avoid many useless derivation subtrees. A much better algorithm is Earley's algorithm [1, 2]:
Let G be a proper CFG. Objects of the form $[A \to X_1 \dots X_k \cdot X_{k+1} \dots X_m, i]$, where $A \to X_1 \dots X_m$ and $0 \leq i \leq n$, are called items. The algorithm builds a table $(T_j)_{1\leq j\leq n}$ such that T_j contains an item $[A \to \alpha \cdot \beta, i]$ iff there exists γ such that

$$S \to^* a_1 \dots a_i A\gamma \to a_1 \dots a_i\alpha\beta\gamma \to^* a_1 \dots a_j\beta\gamma$$

At the beginning, $[S \to \cdot\alpha, 0] \in T_0$ for all $S \to \alpha$. Then, if $[A \to \cdot B\beta, 0] \in T_0$, add $[B \to \cdot\gamma, 0]$ to T_0 for all $B \to \gamma$. For j between 1 and n, perform the following steps

(1) For all $[B \to \alpha \cdot a\beta, i] \in T_{j-1}$ such that $a = a_j$, add $[B \to \alpha a \cdot \beta, i]$ to T_j;
(2) If $[A \to \gamma\cdot, i] \in T_j$, then for all $[B \to \alpha \cdot A\beta, k] \in T_i$, add $[B \to \alpha A \cdot \beta, k]$ to T_j;
(3) If $[A \to \alpha \cdot B\beta, i] \in T_j$, add $[B \to \cdot\gamma, j]$ to T_j for all $B \to \gamma$;
(4) Repeat the two previous steps while there remains items to be added to T_j.

The string $a_1 \dots a_n$ belongs to $L(G)$ iff $[S \to \alpha\cdot, 0] \in T_n$.

All items generated by Earley's algorithm are useful in the context of left to right parse of the input string.

4. Our parsing algorithm

Our parsing algorithm outputs items which are in fact a factorization of Earley's items sharing the same right part before the dot: it will replace a set of items $[A \to \alpha \cdot \beta, i]$ having the same string α by a single item $[\Delta \to \alpha, i]$ if $\alpha \neq \epsilon$, or by nothing if $\alpha = \epsilon$ [8, 5]. Δ is the set of non-terminals which were at the left-hand side of replaced items.

The algorithm replaces the search performed in step 4 of Earley's algorithm by an extraction in a priority queue Q holding pairs (X, i) of symbols and integers. We say that a pair (X, i) has a greater priority than a pair (Y, j) if $i > j$ or if $i = j$ and $Y \to^* X$. The function used to return and remove the set of maximum pairs of Q is denoted by $Extract$.

Let G be a proper CFG. Our algorithm builds a table $(T_j)_{1 \leq j \leq n}$ such that T_j contains an item $[\Delta \to \alpha, i]$ iff $\alpha \neq \epsilon$ and for all $A \in \Delta$ there exists β and γ such that

$$S \to^* a_1 \dots a_i A \gamma \to a_1 \dots a_i \alpha \beta \gamma \to^* a_1 \dots a_j \beta \gamma$$

Every time an item $[\Delta \to \alpha, i]$ is added to T_j, perform $Q := Q \cup \{(A, i) \mid A \in \Delta \wedge A \to \alpha\}$. At the beginning, all T_j are empty. Let $\Delta_0 = \{A \in N \mid \exists \beta, S \to A\beta\}$. For j between 1 and n perform the following steps

(1) $Q := \{(a_j, j-1)\}$;
(2) $(X, i) := Extract(Q)$;
(3) $T_j := T_j \cup \{[\Delta \to X, i] \mid \Delta = \{A \in \Delta_i \mid \exists \beta, A \to X\beta\} \neq \emptyset\}$;
(4) $T_j := T_j \cup \{[\Delta \to \alpha X, h] \mid \exists [\Delta' \to \alpha, h] \in T_i, \Delta = \{A \in \Delta' \mid \exists \beta, A \to \alpha X \beta\} \neq \emptyset\}$;
(5) Repeat steps 2 to 5 while Q is not empty;
(6) Compute $\Delta_j := \bigcup_{[\Delta \to \alpha, i] \in T_j} \{D \in N \mid \exists A \to \alpha B \beta, \exists \gamma, A \in \Delta \wedge B \to^* D\gamma\}$.

Then $a_1 \dots a_n \in L(G)$ iff there exists $[\Delta \to \alpha, 0] \in T_n$ such that $S \in \Delta$ and $S \to \alpha$.

In the general case, the complexity of our algorithm is $O(n^3)$ in time and $O(n^2)$ in space. As with Earley's algorithm, these orders might be improved for grammars having some special properties which are of no interest in our case (RNA folding).

Now that we have an algorithm which may use CFG, we may transform it to use S-ACFG:

- Items are $[\Delta \to \alpha, i, \lambda]$, where λ is the string of attributes attached to α;
- Pairs (X, i) added to Q are triplets (X, i, x), where x is the attribute attached to X;
- Functions $f_{A \to \alpha}$ are taken into account at the time of reduction of items;
- The combinatorial explosion of the number of items is avoided with constraints $\mathcal{C}_A$, associated with non-terminals A, which replace a set of triplets (A, i, x) with fixed A and i by a single triplet (A, i, y) whose attribute y is deduced from attributes in the replaced set.

Let G be a S-ACFG. Every time an item $[\Delta \to \alpha, i, \lambda]$ is added to T_j, perform $Q := Q \cup \{(A, i, f_{A \to \alpha}(\lambda)) \mid A \in \Delta \wedge A \to \alpha\}$. At the beginning, all T_j are empty. Let $\Delta_0 = \{A \in N \mid \exists \beta, S \to A\beta\}$. For $1 \leq j \leq n$ perform the following steps

(1) $Q := \{(a_j, j-1, S_{\mathcal{A}}(a_j))\}$;
(2) $(X, i, x) := \mathcal{C}_X(Extract(Q))$;
(3) $T_j := T_j \cup \{[\Delta \to X, i, x] \mid \Delta = \{A \in \Delta_i \mid \exists \beta, A \to X\beta\} \neq \emptyset\}$;
(4) $T_j := T_j \cup \{[\Delta \to \alpha X, h, \lambda x] \mid \exists [\Delta' \to \alpha, h, \lambda] \in T_i, \Delta = \{A \in \Delta' \mid \exists \beta, A \to \alpha X \beta\} \neq \emptyset\}$;
(5) Repeat steps 2 to 5 while Q is not empty;
(6) Compute $\Delta_j := \bigcup_{[\Delta \to \alpha, i, \lambda] \in T_j} \{D \in N \mid \exists A \to \alpha B \beta, \exists \gamma, A \in \Delta \wedge B \to^* D\gamma\}$.

Then $a_1 \dots a_n \in L(G)$ iff there exists $[\Delta \to \alpha, 0, \lambda] \in T_n$ such that $S \in \Delta$ and $S \to \alpha$.

Let $r \geq 1$ be the maximum number of nonterminals appearing at the right-hand side of any production of G. Then the space complexity is $O(n^r)$ and the time complexity is $O(n^{r+1})$. Grammars

used in practice usually verify $r = 2$ or may be turned into a grammar verifying $r = 2$. Hence our algorithm has the complexity of the dynamic programming algorithm used by Zuker [10] to find a secondary structure of minimal energy with the thermodynamic model.

The main advantage of our algorithm over a simpler Earley algorithm is that, with the factorization provided by our items, many different items may now be replaced by a single item. This feature is interesting with SCFGs we used with our algorithm.

5. Results

We have retrieved by ftp the *Vienna package*. This package is a set of C source files which implements the old style dynamic programming relations popularized by Zuker to find the minimum energy secondary structure of an RNA for the well known thermodynamic model. We then converted the thermodynamic model embedded in this package into a suitable S-ACFG, and then into a C source parser by a YACC-like tool which we wrote. Because they use the same model, our generated parser and the *Vienna package* will return the same secondary structure from the same input, thus we may compare dynamic programming and parsing. On 1667 bases of RNA on a DEC-server 2100-500MP, the Vienna package requires 350 s. and 19 Mbytes, while our program needs 266 s. and 50 Mbytes. Thus our parsing algorithm is faster than standard dynamic programming, and it uses less than three times as much memory. Yet, the description of the thermodynamic model with S-ACFG is much simpler and much more flexible in the expression of structural constraints than dynamic programming relations.

Our parsing algorithm may also be readily applied to SCFGs since probabilities of derivations trees may easily be interpreted as attributes of derivation trees. The first results are encouraging because we are 3 times faster on tRNAs than the CYK-like parser used by Haussler's team.

Bibliography

[1] Aho (Alfred V.) and Ullman (Jeffrey D.). – *The Theory of Parsing, Translation, and Compiling.* – Prentice-Hall, Inc., 1972, vol. 1.

[2] Earley (J.). – An efficient context-free parsing algorithm. *Communication of the ACM*, vol. 13, n° 2, February 1970, pp. 94–102.

[3] Eddy (Sean R.) and Durbin (Richard). – RNA sequence analysis using covariance models. *Nucleic Acids Research*, vol. 22, n° 11, 1994, pp. 2079–2088.

[4] Knuth (Donald E.). – Semantics of context-free languages. *Mathematical Systems Theory*, vol. 2, 1968, pp. 127–145. – Correction: *Mathematical Systems Theory* vol. 5, 1971, pp. 95-96.

[5] Nederhof (Mark-Jan). – A multidisciplinary view on PLR parsing. In *Twentieth Workshop on Language Technology 6.* – December 1993.

[6] Sakakibara (Yasubumi), Brown (Michael), Mian (I. Saira), Sjölander (Kimmen), Underwood (Rebecca C.), and Haussler (David). – *The application of stochastic context-free grammars to folding, aligning and modeling homologous RNA sequences.* – Technical Report n° UCSC-CRL-94-14, University of California, Santa Cruz, Santa Cruz CA 95064 USA, 1993.

[7] Searls (David B.). – The linguistics of DNA. *American Scientist*, vol. 80, 1992, pp. 579–591.

[8] Voisin (F.). – A botttom-up adaptation of Earley's parsing algorithm. In *Programming Languages Implementation and Logic Programming, International Workshop*, pp. 146–160. – Springer-Verlag, May 1988.

[9] Waterman (M. S.). – Secondary structure of single-stranded nucleic acids. *Studies in Foundations and Combinatorics, Advances in Mathematics Supplementary Studies*, vol. 1, 1978, p. 167.

[10] Zuker (Michael). – The use of dynamic programming algorithms in RNA secondary structure prediction. In Waterman (Michael S.) (editor), *Mathematical Methods for DNA Sequences*, Chapter 7, pp. 159–184. – CRC Press, 1989.

Pascal's Triangle, Automata, and Music

Jean-Paul Allouche
Université de Marseille

December 5, 1994

[summary by Philippe Dumas]

The reduction of Pascal's triangle modulo a prime number p, or a power of a prime number, has been intensively studied. It is known that the reduction produces a phenomenon called auto-similarity; natural homotheties with contraction ratio $1/p^k$ appear, and a limit set in the sense of the Hausdorff metric exists; moreover the limit set has fractal dimension $\log\binom{p+1}{2}/\log p$. The reduction modulo a composite number does not produce this phenomenon of auto-similarity; however it is still possible to make a limit set come out [7]. This talk shows a way to describe the complexity of such a double sequence. The basic tool is the concept of a double automatic sequence. First we introduce automatic sequences and complexity of sequences over a finite set; as an illustration it is shown that there are about $(m+n)^2$ distinct rectangular blocks with m rows and n columns in Pascal's triangle reduced modulo a prime number [1]. Next we give a statement about the automaticity of some linear cellular automaton, and specifically of Pascal's triangle reduced modulo an integer [2]. Finally an application to musical composition is mentioned [3, 5].

1. Automatic sequences

Formal language theory provides a way to define infinite words as fixed points of morphisms. As an example, take the alphabet $\{0,1\}$ and the recurrence

$$w_0 = 0, \qquad w_{n+1} = w_n\overline{w_n},$$

where the bar means exchange 0 and 1; the first few terms of the sequence are the following words,

$$\begin{aligned}
w_0 &= 0\\
w_1 &= 01,\\
w_2 &= 0110,\\
w_3 &= 01101001,\\
w_4 &= 0110100110010110,\\
w_5 &= 01101001100101101001011001101001.
\end{aligned}$$

Ultimately an infinite word appears. This word is the Thue-Morse word, which is a fixed point of the substitution [6]

$$\sigma(0) = 01, \qquad \sigma(1) = 10.$$

Another example is defined as follows. Two letters, a left brace { and a right brace } are the elements of the alphabet. The sequence of words A_n is defined by the rules

$$A_0 = \{\}, \qquad A_{n+1} = \{A_0 \dots A_n\}.$$

The limit sequence is a fixed point of

$$\lambda(\{) = \{\{\}, \qquad \lambda(\}) = \}\},$$

and provides the sequence of natural integers, as defined by Bourbaki.

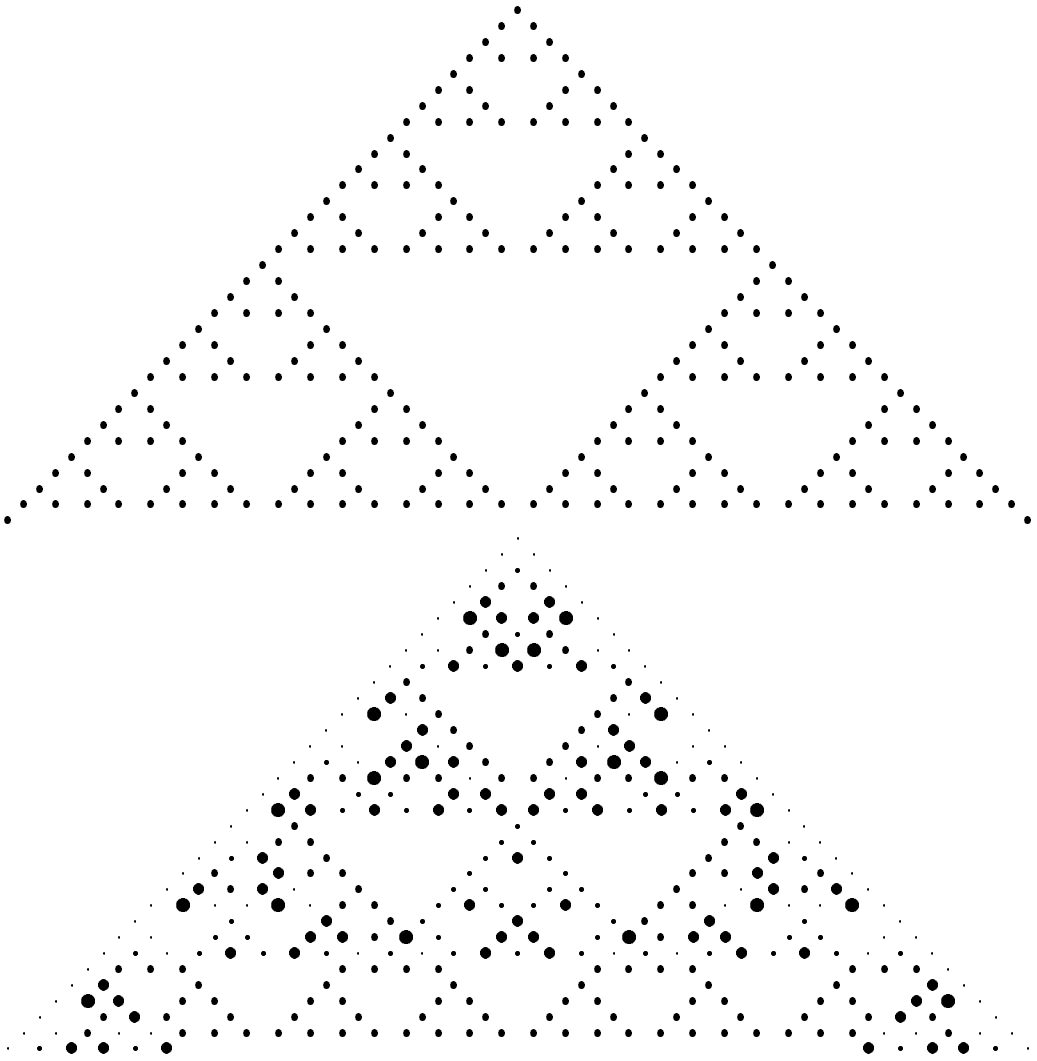


FIGURE 1. Pascal's triangle reduced modulo 2 (top) or modulo 6 (bottom); the size of a dot is proportional to the value of the residue it represents. In the first case the picture is auto-similar, but not in the second one. Nevertheless, in both cases, a limit set exists in the Hausdorff metric.

All these sequences are 2-automatic. (The 2 refers to the fact that the alphabet has two letters.) The definition may be adapted to double sequences. For instance the morphism

$$\beta(0) = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix}, \qquad \beta(1) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$$

applied to the starting point 1 gives Pascal's triangle reduced modulo 2,

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

2. Complexity

The complexity of a sequence over a finite alphabet is defined as another sequence $p(n)$, where $p(n)$ is the number of distinct factors with length n in the given sequence. Obviously the complexity satisfies

$$1 \leq p(n) \leq q^n,$$

if the alphabet is of size q. The complexity reflects how intricate the sequence is. For instance, if for any n the inequality $p(n) \leq n$ is satisfied, the sequence is ultimately periodic. In the case of the Thue-Morse sequence, the sequence of differences $p(n+1) - p(n)$ is 2-automatic. Cobham showed that every automatic sequence has complexity $O(n)$ [4].

For double sequences, the shape of block to use in the definition of complexity is rather arbitrary. A natural choice is to consider rectangular blocks. Then, the complexity is a double sequence $P(m,n)$ where $P(m,n)$ is the number of distinct rectangular blocks with m rows and n columns occurring in the given double sequence.

For Pascal's triangle reduced modulo 2, it is readily noticed that the complexity satisfies

$$P_2(m,n) = P_2(1, m+n-1).$$

Moreover the relation

$$(1+x)^{2t} = (1+x^2)^t$$

shows that row t determines rows $2t$ and $2t+1$; as a consequence the formula

$$P_2(1,n) = n^2 - n + 2$$

is satisfied. More generally, Pascal's triangle reduced modulo a prime p has complexity order n^2. The proof relies on the fact that differences of order 2 of $P_p(1,n)$ form a p-automatic sequence. If the modulus is not a prime but is square-free, for example if the modulus equals 6, the Chinese remainder theorem shows that

$$P_6(1,n) \leq P_2(1,n) P_3(1,n).$$

Actually, the quantities are equal, since the residues modulo 2 and modulo 3 may be considered as independent. More generally, the complexity $P_q(1,n)$ of Pascal's triangle reduced modulo a square free number q is shown to be of order $n^{2\omega(q)}$, where $\omega(q)$ is the number of prime factors of q. The case of prime powers may be tackled by a formula due to Kummer, namely $\left((1+x)^{p^{a-1}}\right)^p = (1+x^p)^{p^{a-1}}$ mod p^a. This result gives a mathematical meaning to the feeling that Pascal's triangle reduced modulo m is more and more complex as the number of prime factors of m increases.

3. Automaticity of linear cellular automata

Pascal's triangle is an example of a linear cellular automaton. There is an initial state, here $g(x) = 1$, and a rule $r(x) = 1 + x$. At time t, the state of the automaton is $g(x)r(x)^t$. To recover a more classical definition from this one, it suffices to consider that coefficients of the state at time t are the contents of cells arranged along the infinite line of integers $\mathbb{Z}$. Moreover the set of states of a cell is finite if the ring of coefficients is finite; here the ring is the ring of integers modulo m. In other words, the double sequence of binomial coefficients reduced modulo m shows the evolution of a classical linear cellular automaton.

It is shown that, when reduced modulo a prime power p^ℓ, a linear cellular automaton provides a p-automatic double sequence. The proof needs an additional concept: a polynomial $r(x)$ is said to have the m-Fermat property if it satisfies

$$r(x^m) = r(x)^m.$$

The Kummer formula above gives an example with $m = p$ a prime number. When the rule $r(x)$ has the m-Fermat property, then the associated double sequence is m-automatic.

As a consequence Pascal's triangle reduced modulo m is m-automatic if m is a prime power. Moreover the converse is true, and its proof relies on Cobham's theorem which asserts that a sequence both p-automatic and q-automatic, p and q being prime and distinct, is ultimately periodic. Here the sequence used is the sequence of central binomials $\binom{2n}{n}$. This result gives a precise formulation of the fact that Pascal's triangle reduced modulo a composite number is more complex than when reduced modulo a prime power.

4. Musical composition

Some composers have used finite automata to produce musical motifs. For instance Tom Johnson has used the morphism defined on a two-letter alphabet $\{+, -\}$ by

$$\mu(+) = + - +, \qquad \mu(-) = - - +.$$

A $+$ codes a melodic ascent, and a $-$ codes a melodic descent. In the same vein, he has used Pascal's triangle reduced modulo 7. The interest of such a composition is that automatic sequences are at the frontier between periodicity and chaos. But as Tom Johnson himself says, this can only be a tool and certainly not a way of composing music in a purely automatic fashion.

Bibliography

[1] Allouche (J.-P.). – Sur la complexité des suites infinies. *Bulletin of the Belgian Mathematical Society*, vol. 1, 1994, pp. 133–143.

[2] Allouche (J.-P.), von Haeseler (F.), Peitgen (H.-O.), and Skordev (G.). – Linear cellular automata, finite automata and Pascal's triangle. – To appear in Discrete Applied Mathematics, 1995.

[3] Allouche (Jean-Paul) and Johnson (Tom). – Finite automata and morphisms in assisted musical composition. – Preprint, 1995.

[4] Cobham (Alan). – Uniform tag sequences. *Mathematical Systems Theory*, vol. 6, n° 2, 1972, pp. 164–192.

[5] Johnson (Tom). – *Formulas for String Quartet.* – Editions 75, 75, rue de la Roquette 75011 Paris, 1994.

[6] Lothaire (M.). – *Combinatorics on Words.* – Addison-Wesley, 1983, *Encyclopedia of Mathematics and its Applications*, vol. 17.

[7] Willson (S.). – Cellular automata can generate fractals. *Discrete Applied Mathematics*, vol. 8, 1984, pp. 91–99.

Riordan Arrays and their Applications

Donatella Merlini
University of Firenze, Italy

October 10, 1994

[summary by Danièle Gardy]

Abstract

A Riordan array is a doubly indexed sequence of coefficients of a bivariate generating function. This talk presents some of their properties, then shows how they can be useful in combinatorial problems.

1. Riordan arrays

The term *Riordan array* was introduced recently to denote a concept familiar in combinatorics; it is a doubly indexed sequence $\{d_{n,k}; n, k \in \mathbb{N}\}$, defined for two formal series $d(t)$ and $h(t)$ by

$$d_{n,k} = [t^n]\{d(t)(th(t))^k\}, \qquad \text{or} \qquad \sum_{n,k} d_{n,k} t^n u^k = \frac{d(t)}{1 - uth(t)}. \tag{1}$$

We use the notation $(d, h) := \{d_{n,k}\}$. A Riordan array is *proper* when $d_{n,n} \neq 0$ for all n, i.e. when $h(0) \neq 0$.

PROPERTY 1. *The $d_{n,k}$ satisfy a recurrence relation*

$$d_{n+1,k+1} = a_0 d_{n,k} + a_1 d_{n,k+1} + \cdots . \tag{2}$$

The a_i's define a series $A(z) = \sum_i a_i z^i$ and the series h satisfies the equation $h(t) = A(th(t))$.

If the recurrence relation (2) holds for some sequence $d_{n,k}$, then this sequence is a proper Riordan array, with $d(t)$ the generating function of the sequence $\{d_{n,0}\}$: $d(t) = \sum_n d_{n,0} t^n$, and $h(t)$ the (unique) solution of the equation $Y = A(tY)$.

THEOREM 1. *Let $f(z) = \sum_k f_k z^k$; then*

$$\sum_k d_{n,k} f_k = [t^n]\{d(t) f(th(t))\}.$$

EXAMPLE. The binomial numbers $\binom{n}{k}$ are defined by $d(t) = h(t) = 1/(1-t)$:

$$\binom{n}{k} = [t^{n-k}]\left\{\frac{1}{(1-t)^{k+1}}\right\} = [t^n]\left\{\frac{t^k}{(1-t)^{k+1}}\right\}.$$

The generating function of the associated sequence $\{a_i\}$ is simply $A(t) = 1 + t$, and the so-called Euler transform is derived from Theorem 1:

$$\sum_k \binom{n}{k} f_k = [t^n]\left\{\frac{1}{1-t} f\left(\frac{1}{1-t}\right)\right\}.$$

2. Combinatorial sums

THEOREM 2. *The Euler transform generalizes as*

$$\sum_k \binom{n+ak}{m+bk} f_k = [t^n]\left\{\frac{t^m}{(1-t)^{m+1}} f\left(\frac{t^{b-a}}{(1-t)^b}\right)\right\} \qquad (b > a),$$
$$= [t^m]\left\{(1+t)^n f\left((1+t)^a t^{-b}\right)\right\} \qquad (b < 0).$$

Theorem 2 applies to sums involving the Catalan numbers $C_k = \binom{2k}{k}/(k+1)$; for example

$$\sum_k \binom{n+k}{m+2k}(-1)^k C_k = \binom{n-1}{m-1}.$$

It does not apply directly to Stirling numbers of the first and second kind $\left[{n \atop k}\right]$ and $\left\{{n \atop k}\right\}$; however the simple identities

$$\sum_n \frac{k!}{n!}\left[{n \atop k}\right] t^n = \log^k \frac{1}{1-t}; \qquad \sum_n \frac{k!}{n!}\left\{{n \atop k}\right\} t^n = (e^t - 1)^k$$

allow us to use a modified form of it. For example, Theorem 2 after some algebra gives

$$\sum_k \left\{{m \atop k}\right\}\left[{k+1 \atop p}\right]\frac{(-1)^{k-p+1}}{k+1} = \frac{1}{m+1}\binom{m+1}{p} B_{m-p+1},$$

where B_n is a Bernoulli number.

3. Inversion formulæ

The *product* of two Riordan arrays $D = (d(t), h(t)) = \{d_{n,k}\}$ and $F = (f(t), g(t)) = \{f_{n,k}\}$ is the double sequence $\{g_{n,k} = \sum_j d_{n,j} f_{j,k}\}$.

PROPERTY 2. *The product of two Riordan arrays is a Riordan array:*

$$D \cdot F = (d(t) f(th(t)), h(t) g(th(t))).$$

The identity element is $I = (1, 1)$.

PROPERTY 3. *A proper Riordan array $D = (d(t), h(t))$ has an inverse $d^{-1} = \{\bar{d}_{n,k}\} = (\bar{d}(t), \bar{h}(t))$.*

As a consequence, we obtain an inversion formula for sums:

$$\sum_k d_{n,k} f_k = g_n \iff \sum_k \bar{d}_{n,k} g_k = f_n.$$

Such formulæ date back a long time; see for example Riordan's book [9]. The inversion formulæ in this book were recently revisited by Sprugnoli [11], who proved them again using the theory of Riordan arrays; see also [10].

4. Coloured walks

We consider three types of steps on a square lattice: North (N), East (E) and North-East (NE). An underdiagonal walk is entirely below or on the main diagonal $x = y$. A *weakly underdiagonal walk* is a walk such that its final point is on or below the diagonal. Let $p_{n,k}$ and $q_{n,k}$ be respectively the number of underdiagonal walks and the number of weakly underdiagonal walks, with n steps and ending at a distance k from the diagonal; the distance k can be either along the x-axis or along the y-axis.

If there is only one kind of step of each type, then we have a Motzkin walk, corresponding to Motzkin words. A generalization allows for different kinds of horizontal (E), vertical (N) or diagonal (NE) steps [7]. Let a, b and c be the number of different steps in the East, North-East and North directions; then we have the following result.

THEOREM 3. *The $\{p_{n,k}\}$ and $\{q_{n,k}\}$ are Riordan arrays such that the associated A function is $A(t) = a + bt + ct^2$, and, with $\Delta = 1 - 2bt + (b^2 - 4ac)t^2$,*

$$\{p_{n,k}\} = \left(\frac{1 - bt - \sqrt{\Delta}}{2act^2}, \frac{1 - bt - \sqrt{\Delta}}{2ct^2}\right);$$

$$\{q_{n,k}\} = \left(\frac{1}{\sqrt{\Delta}}, \frac{1 - bt - \sqrt{\Delta}}{2ct^2}\right).$$

Symmetric walks. When there is the same number of colours for East and North steps ($a = c$), then it is possible to derive some interesting identities. For example, let $\{f_k\}$ be the periodic sequence $\{1, 0, -1, 0, 1, 0, -1, \ldots\}$; then $\sum_k p_{n,k} f_k = b^n$. The algebraic proof of this equality is easy; there also exists a combinatorial interpretation: There is a bijection between the underdiagonal walks ending at distance k, whose last non-NE step is N, and the walks ending at a distance $k + 2$ whose last non-NE step is E.

5. Asymptotics for convolution matrices

The reference for this section is [8]. Let F be an analytic function s.t. $F(0) = 1$, and define $F_n(x) = [z^n]\{F(z)^x\}$. This is a polynomial function of degree n, satisfying a convolution property:

$$F_n(x + y) = \sum_{k=0}^{n} F_{n-k}(x) F_k(y).$$

The $f_{n,k} = [x^k]\{n! F_n(x)\} = [x^k z^n]\{n! F(z)^x\}$ define an infinite *convolution matrix* [1, 6]. Let $d_{n,k} := \frac{k!}{n!} f_{n,k}$; then

$$d_{n,k} = \sum \frac{k!}{k_1! k_2! k_3! \cdots} \left(\frac{f_1}{1!}\right)^{k_1} \left(\frac{f_2}{2!}\right)^{k_2} \left(\frac{f_3}{3!}\right)^{k_3} \cdots,$$

which shows that $\{d_{n,k}\}$ is a Riordan array: $\{d_{n,k}\} = (1, (\ln F(z))/z)$. In other terms, $f_{n,k} = (n!/k!)[z^n]\{(\ln F(z))^k\}$.

For a fixed ratio $p = n/k$, we have an asymptotic equivalent of $d_{n,k}$, or equivalently of $f_{n,k}$: Define $m = n - k$ and

$$\Phi_p(z) = \left(\frac{\ln F(z)}{z}\right)^{1/(p-1)} = \left(\frac{\ln F(z)}{z}\right)^{\frac{k}{m}};$$

then $d_{n,k} = [z^m]\{\Phi_p(z)^m\}$. As long as p is fixed, we can define $C(u) = \sum_m [z^m]\{\Phi_p(z)^m\} u^m$. A form of the function $C(u)$ can be computed as follows. There exists a unique analytical function

$w(z)$ s.t. $w(0) = 0$ and $w(z) = z\Phi_p(w(z))$. Define a function G by $G'(u) = 1/\Phi_p(u)$; then, by the Lagrange Inversion Formula,

$$[z^n]\{G(w(z))\} = \frac{1}{n}[u^{n-1}]\left\{G'(z)\Phi_p(u)^n\right\} = \frac{1}{n}[u^{n-1}]\left\{\Phi_p(u)^{n-1}\right\}.$$

Applying this formula backwards to $[z^m]\{\Phi_p(z)^m\}$ gives

$$d_{n,k} = [u^m]\{C(u)\} \qquad \text{with} \qquad C(u) = \frac{uw'(u)}{w(u)} = \frac{1}{1 - u\Phi_p'(w(u))}.$$

When the function Φ_p is well-behaved, an asymptotic equivalent of $d_{n,k}$ can be obtained by singularity analysis. For example, assume that the radius of convergence of $w(z)$ is finite, and that w has a single singularity r on its circle of convergence, defined as the solution of smallest modulus of the equation $z\Phi_p'(w(z)) = 1$. Let $s = w(r)$; as $s = r\Phi_p(s)$, we get that $s\Phi_p'(s) = \Phi_p(s)$. This leads finally to the asymptotic formula

$$d_{n,k} \sim \frac{\Phi_p'(s)}{\sqrt{2\Phi_p(s)\Phi_p''(s)}} \frac{\Phi_p'(s)^m}{4^m} \binom{2m}{m}. \tag{3}$$

If desired, this equivalent can be expanded into an asymptotic expansion to any order.

EXAMPLE. Asymptotic estimates for Stirling numbers of the second kind have been obtained by several authors (see for example [12] for a survey of results and for uniform expansions); some estimates can also be derived from (3): For $f_{n,k} = \left\{ {n \atop k} \right\}$, we have $d_{n,k} = (k!/n!)\left\{ {n \atop k} \right\}$ and $\Phi_p(z) = h(z)^{1/(p-1)}$ with $h(z) = (e^z - 1)/z$. The equation defining s simplifies into $e^s = p/(p-s)$. Then $\Phi_p(s) = 1/(p-s)^{k/m}$, $\Phi_p'(s) = \Phi_p(s)/s$ and $\Phi_p''(s) = \Phi_p(s)(p(s-p+1))/(s^2(p-1))$. The asymptotic equivalent (3) gives after some computation

$$\begin{aligned}\left\{ {n \atop k} \right\} &\sim \frac{n!}{k!}\sqrt{\frac{\Phi_p(s)}{2s^2\Phi_p''(s)}} \binom{2n-2k}{n-k} \left(\frac{\Phi_p(s)}{4s}\right)^m \\ &\sim \frac{n!}{k!}\sqrt{\frac{k(n-k)}{2n(ks-n+k)}} \binom{2n-2k}{n-k} \frac{k^k}{(4s)^{n-k}(n-sk)^k}.\end{aligned}$$

Numerical results for Stirling numbers of the second kind are then presented for given n and k, and hence p; the conditions of application of the formula (3) are not satisfied (the formula holds for *constant* p and $n \to +\infty$) but the approximations computed are very close to the actual values, which suggests that the range of application of (3) is much wider than indicated, and that some kind of uniformity result should hold.

Indeed, for a fixed ratio $p = n/k$, the asymptotic expansion given by (3) can also be obtained by a saddle-point approximation. We give here the computation for the first term of the expansion; the full asymptotic expansion can probably be obtained in a similar way. For $n - k = m \to +\infty$, we have to compute $[z^m]\Phi_p^m(z)$. The saddle-point ρ_0 is defined by the equation $z\Phi_p'(z)/\Phi_p(z) = 1$; the uniqueness of the solution on $]0, +\infty[$ shows that $\rho_0 = s$. Then

$$[z^m]\{\Phi_p^m(z)\} \sim \frac{\Phi_p(s)^m}{s^m\sqrt{2\pi m\sigma^2}} \qquad \text{with} \qquad \sigma^2 = s^2\left(\frac{\Phi_p''}{\Phi_p}(s) - \frac{\Phi_p'^2}{\Phi_p^2}(s) + \frac{\Phi_p'(s)}{s\Phi_p(s)}\right). \tag{4}$$

As $\Phi_p^{'}(s)/\Phi_p(s) = 1/s$ here, σ^2 is simply $s^2\Phi_p^{''}(s)/\Phi_p(s)$. Injecting this into Equation (4), and with $\Phi_p(s) = s\Phi_p^{'}(s)$, we get

$$[z^m]\{\Phi_p^m(z)\} \sim \frac{\Phi_p^{'}(s)^{m+1}}{\sqrt{2\pi m \Phi_p(s)\Phi_p^{''}(s)}},$$

which is exactly (3) if we use Stirling's approximation for the factorial: $\binom{2m}{m}4^{-m} \sim 1/\sqrt{\pi m}$.

Thus, the approach presented in this talk can be seen as an alternative to the saddle-point approach; instead of solving the equation $s\Phi_p^{'}(s) = \Phi_p(s)$, it leads to solving the equation $w(r) = r\Phi_p(w(r))$, which may be simpler in some cases.

To show how we can obtain asymptotic expansions for a large range of n and k, we write

$$[z^m]\{\Phi_p^m(z)\} = [z^m]\left\{f(z)^k\right\} \qquad \text{with} \qquad f(z) = \frac{\log F(z)}{z}.$$

Now $f^k(z) = \Phi_p(z)^m$ and the saddle-point approximation gives

$$[z^m]\{f^k(z)\} \sim \frac{f(\rho_1)^k}{\sigma\rho_1^m\sqrt{2\pi k}}, \tag{5}$$

for ρ_1 the (unique) real positive solution of the equation $zf^{'}(z)/f(z) = m/k$ and

$$\sigma^2 = \rho_1^2((f^{''}/f)(\rho_1) - (f^{'2}/f^2)(\rho_1) + (f^{'}(\rho_1)/\rho_1 f(\rho_1)).$$

Now $f^{'}/f = (m/k)(\Phi_p^{'}/\Phi_p)$ and the saddle-point is still $\rho_1 = s$; also $\sigma^2 = (m/k)\rho_1^2(\Phi_p^{''}/\Phi_p)(\rho_1)$; hence the equation (5) is simply another way of writing (3) or (4). However, in this last form, it is easy to understand why the approximation (3) holds for n/k no longer fixed: The equivalent approximation (5) has been proved for $m = \Theta(k)$ [2, 5] or for $m = o(k)$ [3, 4]. This indicates that the asymptotic expansion (3) is valid without restriction on $p = n/k$, as long as $n = k + O(k)$, and $m = n - k \to +\infty$.

Bibliography

[1] Carlitz (L.). – A special class of triangular arrays. *Collectanea Mathematica*, vol. 27, 1976, pp. 23–58.

[2] Daniels (H. E.). – Saddlepoint approximations in statistics. *Annals of Mathematical Statistics*, vol. 25, 1954, pp. 631–650.

[3] Drmota (M.). – A bivariate asymptotic expansion of coefficients of powers of generating functions. *European Journal of Combinatorics*, vol. 15, 1994, pp. 139–152.

[4] Gardy (D.). – Some results on the asymptotic behaviour of coefficients of large powers of functions. *Discrete Mathematics*, 1995.

[5] Good (I. J.). – Saddle-point methods for the multinomial distribution. *Annals of Mathematical Statistics*, vol. 28, 1957, pp. 861–881.

[6] Knuth (D. E.). – Convolution polynomials. *The Mathematica Journal*, vol. 2, 1992, pp. 67–78.

[7] Merlini (D.), Sprugnoli (R.), and Verri (M. C.). – Algebraic and combinatorial properties of simple, coloured walks. In *CAAP, Lecture Notes in Computer Science*, vol. 787, pp. 218–233. – 1994.

[8] Merlini (D.), Sprugnoli (R.), and Verri (M. C.). – Asymptotics for two-dimensional arrays: convolution matrices. – June 1994.

[9] Riordan (J.). – *Combinatorial identities*. – Wiley, New York, 1968.

[10] Sprugnoli (R.). – Riordan arrays and combinatorial sums. *Discrete Mathematics*, 1994.

[11] Sprugnoli (R.). – A unitary approach to combinatorial inversions. – June 1994.

[12] Temme (N. M.). – Asymptotic estimates of Stirling numbers. *Studies in Applied Mathematics*, vol. LXXXIX, n° 3, 1993, pp. 233–244.

Structured Numbers

Vincent Blondel
INRIA Rocquencourt

April 10, 1995

[summary by Philippe Flajolet]

Abstract

The talk describes a "lifting" of the system of unary representations of numbers into a system of tree-like representations. Alternatively, this can be seen as an arithmetic description of certain combinatorial properties of trees. In particular, addition, multiplication, and exponentiation of trees can be defined in a natural way.

This talk is based on [1]. We start with the family of *complete binary trees* [3], where each node has either 0 or 2 successors. The trees considered are rooted and embedded in the plane, so that left and right are distinguished. Nodes without successors are the external nodes, sometimes called leaves. The *weight* or size of a tree t is taken to be the number of its external nodes and is denoted by $|t|$. It has been well-known for over a century (bracketing problems, see [2]) that the number of trees of size n is given by the Catalan number,

$$T_n = \frac{1}{n}\binom{2n-2}{n-1}. \tag{1}$$

One may well view a tree of size n as a tree-like representation of integer n, and try to generalize the usual operations of addition, multiplication, and so on, of the integers. In other words, we also regard trees as extending the unary representation of integers with some supplementary structure superimposed, hence the name of "*structured numbers*" in the title.

1. Operations

First, *addition* is the basic operation defined as associating to two trees, u and v, the tree

$$u + v := (u \cdot v)$$

obtained by taking a root node and appending u and v to it, as left and right subtrees respectively. Given that weight is defined by number of external nodes, one has $|t| = |u| + |v|$, so we do capture in a way the usual addition of integers. On the other hand, it is clear that addition of structured numbers is not in general commutative.

Multiplication should be defined as a suitable iteration of addition, exponentiation as an iteration of multiplication, *etc.* Such a process is taking its inspiration from what has been done for corresponding integer operations; in that case, a denumerable collection of operations result that lead to the classical Ackermann function of recursive function theory. The talk and the paper [1] both propose to examine what survives of properties of integers in this context.

A whole *hierarchy* of binary operations on trees is introduced recursively as follows:

$$a \cdot^1 b = a + b, \qquad a \cdot^{k+1} b = (a \cdot^k b_0) \cdot^k (a \cdot^k b_1),$$

where b_0, b_1 are the left and right root subtrees of b. We thus have by definition $a \cdot^1 b = a + b$, and we define *multiplication* and *exponentiation* by

$$a \times b = a \cdot^2 b, \qquad a^b = a \cdot^3 b.$$

Note that the multiplication in $a \times b$ can be viewed as the process of grafting copies of a at each leaf of b, that is to say, as the substitution $b[a]$.

THEOREM 1 (WEIGHT THEOREM). *With a and b arbitrary trees, one has*

$$|a + b| = |a| + b|, \quad |a \times b| = |a| \cdot |b|, \quad |a^b| = |a|^{|b|}.$$

For higher-order operations, the weight of the result is no longer independent of the shape of the operands.

THEOREM 2 (DISTRIBUTIVITY). *With a, b, c arbitrary trees, one has*

$$a \cdot^k (b + c) = (a \cdot^k b) \cdot^{k-1} (a \cdot^k c), \quad a \cdot^k (b \times c) = (a \cdot^k b) \cdot^k c.$$

For instance, we have the natural generalizations

$$a \times (b + c) = a \times b + a \times c, \; a \times (b \times c) = (a \times b) \times c,$$
$$a^{(b+c)} = a^b \times a^c, \; a^{b \times c} = (a^b)^c.$$

These two theorems are representative of the results of [1]. Other properties include right simplifiability of $+, \times$ (the property stops at level 3 of the hierarchy!).

2. Prime trees

It is clear from the interpretation of multiplication as substitution that a tree is composite or non-prime (is non-trivially decomposable under multiplication) if and only if one of its "fringes" consists of identical trees. Each tree then factors uniquely into primes [1]. From there, it is natural to ask whether there is some sort of a prime density theorem for structured numbers. The answer was obtained jointly by the speaker and the author of this summary. We briefly explain it here.

The number T_n of all trees of size n is known and given by (1). Let I_n be the number of those that are primes; clearly, we must adopt $I_1 = 0$ (1 is not a prime!). Let $T(z)$, $I(z)$ be the corresponding generating functions:

$$T(z) = \sum_{n \geq 1} T_n z^n = z + z^2 + 2z^3 + 5z^4 + 14z^5 + 42z^6 + \cdots,$$

$$I(z) = \sum_{n \geq 1} I_n z^n = z^2 + 2z^3 + 4z^4 + 14z^5 + 38z^6 + \cdots.$$

Combinatorial classics [2] teach us that

$$T(z) = \frac{1 - \sqrt{1 - 4z}}{2}.$$

Now decomposing trees according to their prime "trailers" yields a relation defining $I(z)$ implicitly:

$$T(z) = z + \sum_{k=2}^{\infty} T_k I(z^k), \qquad T_n = \delta_{n,1} + \sum_{d|n, d \geq 2} T_{n/d} I_d. \tag{2}$$

We recognize here a product of (formal) Dirichlet series. Setting

$$\tau(s) = \sum_{n=1}^{\infty} \frac{T_n}{n^s}, \qquad \iota(s) = \sum_{n=1}^{\infty} \frac{I_n}{n^s},$$

we have the relation matching (2):

$$\tau(s) = 1 + \tau(s)\iota(s) \quad \text{or} \quad \iota(s) = 1 - \frac{1}{\tau(s)}.$$

Thus expanding $1/\tau(s)$ as $(1 + v(s))^{-1}$ yields

$$I_n = T_n - \sum_{\substack{d_1 d_2 = n \\ d_j \geq 2}} T_{d_1} T_{d_2} + \sum_{\substack{d_1 d_2 d_3 = n \\ d_j \geq 2}} T_{d_1} T_{d_2} T_{d_3} - \cdots .$$

In particular $T_n - I_n$ is equal to 0 if n is prime (as it should), is equal to $(T_p)^2$ if $n = p^2$ is the square of a prime, and is otherwise approximated by $2T_p T_{n/p}$ if p is the smallest prime divisor of n. Here are a few initial values.

n	1	2	3	4	5	6	7	8	9	10	11	12
T_n	1	1	2	5	14	42	132	429	1430	4862	16796	58782
I_n	0	1	2	4	14	38	132	420	1426	4834	16796	58688

Note that the asymptotic form of T_n results from Stirling's formula:

$$T_n \sim \frac{4^{n-1}}{\sqrt{\pi n^3}}.$$

Clearly, almost trees are irreducible: the asymptotic density of primes is thus 1 and further characterized by the remarks above.

Bibliography

[1] Blondel (Vincent). – Une famille d'opérateurs sur les arbres binaires. *Comptes-Rendus de l'Académie des Sciences*, vol. 321, 1995, pp. 491–494.

[2] Comtet (Louis). – *Advanced Combinatorics.* – Reidel, Dordrecht, 1974.

[3] Knuth (Donald E.). – *The Art of Computer Programming.* – Addison-Wesley, 1968, vol. 1: Fundamental Algorithms. Second edition, 1973.

Part 2

Symbolic Computation

Evaluating Signs of Determinants Using Single-Precision Arithmetic

Jean-Daniel Boissonnat
INRIA Sophia-Antipolis

May 15, 1995

[summary by Brigitte Vallée, Université de Caen]

Abstract

Most decisions in geometric algorithms are based on signs of determinants. For example, deciding if a point belongs to a given half-space or a given ball reduces to evaluating the sign of a determinant. It is therefore crucial to have reliable answers to such tests and to produce robust algorithms. There exist basically two categories of approaches to this objective:

- rounded computations, followed by a proof of the topological correctness of the result;
- exact integer computations that use $n\ell$ bits for the computation of an $n \times n$ determinant with ℓ-bit integers as inputs.

Here, the second approach is followed, the goal being to use as few bits as possible to evaluate signs of determinants. For dimensions $n = 2$ and $n = 3$, the algorithms proposed require respectively ℓ and $\ell + 1$ bits arithmetic, and run in polynomial time in ℓ: they perform in the worst case respectively $O(\ell)$ and $O(\ell^3)$ elementary operations —additions, subtractions, comparisons, and Euclidean divisions— on integers of ℓ or $\ell + 1$ bits. Extensive simulations have shown that the algorithms perform well in practice so that the average-case complexity appears to be much better than the worst-case complexity. This observation can be proved in the two-dimensional case [6]. Under heuristic hypotheses, the proof can be extended to the three-dimensional case [5].

This talk is based on a joint paper of Francis Avnaim, Jean-Daniel Boissonnat, Olivier Devillers, Franco P. Preparata, and Mariette Yvinec [1]. The author of the summary has interpreted some ideas of the original lecture and has proven a conjecture stated there [5, 6].

1. Two-dimensional case.

The aim is to evaluate the sign of a 2×2 nontrivial determinant,

$$D = \det \begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \end{pmatrix},$$

with nonzero integer entries of at most ℓ bits. By dividing the first column by x_1 and the second column by y_1, one can write $D = x_1 y_1 D'$ with

$$D' = \det \begin{pmatrix} 1 & 1 \\ x & y \end{pmatrix} = y - x,$$

where $y = y_2/y_1$ and $x = x_2/x_1$. Evaluating the sign of a 2×2 determinant thus reduces to evaluating the sign of the difference between two rationals x and y. We consider the set $\mathcal{J}_\ell$ of rationals in the interval $]0, 1/2]$ whose numerator and denominator have at most ℓ bits.

The main idea is to expand both rationals x and y into continued fractions: the comparison between both expansions (under lexicographic order) suffices to compare the rationals themselves. The outline of the algorithm is thus very simple:

> As long as x and y have matching continued fractions expansions, continue expanding; stop as soon as the expansions differ.

There are two variants of the algorithm, which depend on the kind of continued fraction that is used: The *Standard-Sign* algorithm is based on standard continued fractions that are built with the usual Euclidean division $a = bq + r$ with $0 \leq r < b$ while the *Centered-Sign* algorithm is based on centered continued fractions that are built with the centered Euclidean division $a = bq + r$ with $|r| \leq b/2$. The worst-case of these algorithms arises when x and y are equal and the analysis uses well-known results of Lamé (1845) [4] and Dupré (1846) [3] relative to the standard and centered gcd algorithm respectively. The average number of iterations is quite different from the worst case since it is asymptotically constant (i.e., independent of the number ℓ of bits of the input) [6]. Not too surprisingly similar constants show up in the average-case analysis of lattice reduction algorithms in the two-dimensional case [2].

THEOREM 1. *On rationals x and y of $\mathcal{J}_\ell$, the algorithms perform a number of iterations L at most*

$$\begin{cases} \ell \frac{\log 2}{\log(1+\sqrt{2})}, & \text{for the Centered-Sign algorithm,} \\ \ell \frac{\log 2}{\log \phi}, & \text{for the Standard-Sign algorithm.} \end{cases}$$

(Here, ϕ is the golden ratio equal to $\phi = (1+\sqrt{5})/2$). If the entries x and y are taken in the square $\mathcal{J}_\ell \times \mathcal{J}_\ell$ with a density $F(x,y)$ proportional to $|x-y|^r$ (with $r > -1$), the average number of iterations $E[L]$ of the Centered-Sign algorithm is asymptotic to

$$\beta(r) = \frac{4}{\zeta(4+2r)} \sum_{d=1}^{\infty} \frac{1}{d^{2+r}} \sum_{c=\lceil d\phi \rceil}^{\lfloor d\phi^2 \rfloor} \frac{1}{c^{2+r}}, \qquad \ell \to \infty,$$

and the average number of iterations $E[L]$ of the Standard-Sign algorithm is asymptotic to

$$\alpha(r) = \frac{4}{\zeta(4+2r)} \sum_{d=1}^{\infty} \frac{1}{d^{2+r}} \sum_{d < c \leq 2d} \frac{1}{c^{2+r}}, \qquad \ell \to \infty.$$

In particular, when the density F is uniform, the average numbers of iterations are respectively asymptotic to

$$\beta := \beta(0) = \frac{4}{\zeta(4)} \sum_{d=1}^{\infty} \frac{1}{d^2} \sum_{c=\lceil d\phi \rceil}^{\lfloor d\phi^2 \rfloor} \frac{1}{c^2} = 1.08922\ldots,$$

$$\alpha := \alpha(0) = \frac{4}{\zeta(4)} \sum_{d=1}^{\infty} \frac{1}{d^2} \sum_{d < c \leq 2d} \frac{1}{c^2} = 1.20226\ldots.$$

In Fig. 1, the domains $[L = k]$ relative to the Standard-Sign algorithm are represented alternatively in black (for odd values of k) and white (for even values of k).

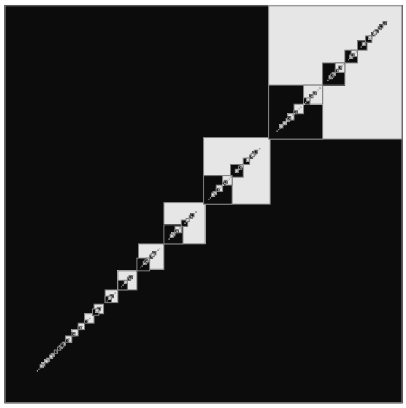


FIGURE 1. The domains $[L = k]$ relative to the Standard-Sign algorithm.

2. Three-dimensional case.

Let D denote a 3×3 determinant with V_1, V_2, V_3 as row vectors. The components x_i, y_i, z_i of vector V_i are assumed to be ℓ-bit integers. The vertical components of the input vectors play a special rôle in the algorithm: all the projections performed are projections parallel to E_z where E_z denotes the last vector of the canonical basis of $\mathbb{R}^3$. Using properties of the determinant, one can assume without loss of generality that the vertical components z_i are positive and in increasing order.

The sign of D depends on which side vector V_3 lies with respect to the plane $\mathcal{P} := \langle V_1, V_2 \rangle$ and this sign is more difficult to evaluate if vector V_3 is very "close" to plane $\mathcal{P}$. We thus define a neighbourhood $\mathcal{V}$ of the plane $\mathcal{P}$ that satisfies the following two properties:

- (a) If V_3 does not belong to $\mathcal{V}$, then it is easy to determine on which side vector V_3 lies with respect to the plane $\mathcal{P}$, the problem reducing to evaluating the sign of a 2×2 determinant;
- (b) If V_3 belongs to $\mathcal{V}$, then there exists a vector W_3 obtained by translating V_3 parallely to $\mathcal{P}$ whose last component z'_3 satisfies $|z'_3| \leq z_3/2$; the algorithm then continues with the new vector system (V_1, V_2, W_3).

The vector V_3 decomposes as $V_3 = \lambda_1 V_1 + \lambda_2 V_2 + \rho E_z$, with rational components $\lambda_1, \lambda_2, \rho$. The numerators and denominators of these rationals may have 2ℓ bits, so that we cannot directly operate with them. The determinant D satisfies

$$D := \det(V_1, V_2, V_3) = \rho \det(V_1, V_2, E_z) = \rho \det(v_1, v_2),$$

where v_i is the projection of V_i on the horizontal plane $z = 0$. If we evaluate the sign of the rational ρ —without explicitly computing it— the problem reduces to evaluating the sign of the determinant $\det(v_1, v_2)$, which is of order 2.

Let $\mathcal{R}$ be the lattice generated by the vectors V_1 and V_2. The fundamental centered parallelogram $\mathcal{F}$ of lattice $\mathcal{R}$ is defined as the set $\mathcal{F} := \{V = \mu_1 V_1 + \mu_2 V_2; |\mu_i| \leq 1/2\}$. The parallelogram $\mathcal{F}$ divides into four sub-parallelograms $\mathcal{F}_i$ that correspond to the four quadrants determined by the four possible signs of (μ_1, μ_2).

Let V be the vector of lattice $\mathcal{R}$ for which $V_3 - V$ is projected inside the fundamental centered

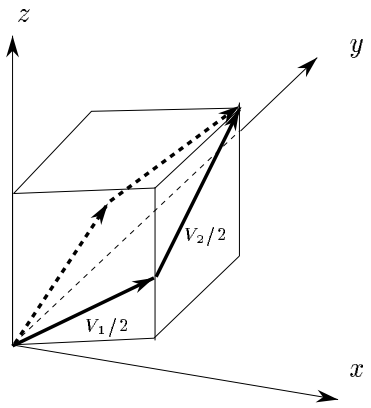


FIGURE 2. The sub-box $\mathcal{B}_1$.

parallelogram $\mathcal{F}$. The integer vector V decomposes as $V := \lfloor\lambda_1\rceil V_1 + \lfloor\lambda_2\rceil V_2$ where $\lfloor\lambda\rceil$ denotes the integer nearest to rational λ. The integer vector $V_3' := V_3 - V$ is $V_3' = \rho_1 V_1 + \rho_2 V_2 + \rho E_z$. Here, the components ρ_i are rationals with absolute value less than $1/2$ and the integer z_3' denotes the vertical component of V_3'. The vector $\rho_1 V_1 + \rho_2 V_2$ is thus a vector of plane $\mathcal{P}$ with rational vertical component $z' := \rho_1 z_1 + \rho_2 z_2$. Since $\rho = z_3' - z'$, the sign of ρ is easy to evaluate provided that we can evaluate the sign of the difference between the integer z_3' and the rational z'. This is the case when the vector V_3' does not belong to the box $\mathcal{B}$ which is defined as follows: The elementary box $\mathcal{B}$ is the union of the four sub-boxes $\mathcal{B}_i$; the sub-box $\mathcal{B}_i$ is a cylinder of direction E_z, of basis $\mathcal{F}_i$, which is delimited by two horizontal planes, whose equations are:

$$\begin{cases} z = 0 \quad \text{and} \quad z = (z_1 + z_2)/2, & \text{for } i = 1; \\ z = -z_1/2 \quad \text{and} \quad z = z_2/2, & \text{for } i = 2; \\ z = -(z_1 + z_2)/2 \quad \text{and} \quad z = 0, & \text{for } i = 3; \\ z = -z_2/2 \quad \text{and} \quad z = z_1/2, & \text{for } i = 4. \end{cases}$$

Thus, each sub-box has an height equal to $(z_1 + z_2)/2$; the sub-box $\mathcal{B}_1$ is represented in Fig 2. The neighborood $\mathcal{V}$ of plane $\mathcal{P}$ is defined as the union of all boxes of the lattice $\mathcal{R}$ obtained by translation of $\mathcal{B}$ by a vector of lattice $\mathcal{R}$.

If the vector V_3 belongs to the neighbourhood $\mathcal{V}$, the vector V_3' belongs to the box $\mathcal{B}$ and different cases are to be considered:

- If the vector V_3' is projected inside $\mathcal{F}_i$ for $i = 2$ or $i = 4$, we let $W_3 := V_3'$; the absolute value of the last component z_3' of vector W_3 is less than $z_2/2$, and the other components of vector W_3 have always at most ℓ bits; the algorithm continues with the system (V_1, V_2, W_3).
- If the vector V_3' is projected inside $\mathcal{F}_i$ for $i = 1$ or $i = 3$, two sub-cases are to be considered: $|\rho_1| \leq |\rho_2|$ and $|\rho_1| \geq |\rho_2|$. The vector W_3 is then defined as follows: W_3 is the vector among the two vectors V_3' or $V_2 - V_3'$ whose last component has the smaller modulus (in the first case). W_3 is the vector among the two vectors V_3' or $V_1 - V_3'$ whose last component has the smaller modulus (in the second case). In both cases, the last component of vector W_3 is in absolute value less than $z_2/2$ and its other components have always at most ℓ bits; the algorithm continues with the system (V_1, V_2, W_3).

We give now the precise description of the algorithm.

Preliminary step. Order the vectors V_1, V_2, V_3 and, if necessary, change some of them into their opposite, in such a way that the vertical components (z_1, z_2, z_3) are positive and sorted in increasing order.

While $z_2 \neq 0$ **do**

1. Compute the vector V of lattice $\mathcal{R}$. Let $V_3' := V_3 - V$.
2. **If** V_3' does not belong to box $\mathcal{B}$
   then evaluate the sign of $\det(v_1, v_2)$ and exit;
   else compute the vector W_3; $V_3 := W_3$. Order the vectors V_1, V_2, V_3 and, if necessary, change them into their opposite so that their vertical components are positive and in increasing order.

At each iteration, either the algorithm evaluates the sign of ρ (if the test in step 2 is positive) and then terminates by the evaluation of a 2×2 determinant, or it continues iteratively on a 3×3 determinant where the largest of the last components has been divided by at least two, the other components remaining unchanged. Thus, the number of iterations of the algorithm is at most equal to 3ℓ.

At each iteration, one computes the nearest integers to rational numbers λ_1 and λ_2,

$$\lambda_1 = \frac{\det \begin{pmatrix} x_3 & y_3 \\ x_2 & y_2 \end{pmatrix}}{\det \begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \end{pmatrix}}, \qquad \lambda_2 = \frac{\det \begin{pmatrix} x_1 & y_1 \\ x_3 & y_3 \end{pmatrix}}{\det \begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \end{pmatrix}}.$$

These rational numbers are quotients of two determinants of order 2 having ℓ-bit integer entries, and cannot be computed directly in single precision. The nearest integers $\lfloor \lambda_i \rceil$ are evaluated, bit by bit by means of a dichotomic process that uses the signs of determinants

$$\det \begin{pmatrix} x_3 - kx_1 & y_3 - ky_1 \\ x_2 & y_2 \end{pmatrix} \quad \text{or} \quad \det \begin{pmatrix} x_3 - kx_2 & y_3 - ky_2 \\ x_1 & y_1 \end{pmatrix}, \qquad |k| = 1, 2, 4, \ldots, 2^\ell,$$

with entries of at most $\ell + 1$ bits. Thus, the computation of vector V uses at most 4ℓ evaluations of signs of 2×2 determinants with entries of at most $\ell + 1$ bits.

THEOREM 2. *Let D be a 3×3 determinant with ℓ-bit integer entries. The determinant sign algorithm above performs at most 3ℓ iterations in the worst-case, each iteration involving the evaluation of at most $4\ell + 9$ signs of determinants 2×2 with $\ell + 1$–bit integer entries. In the worst-case, the algorithm requires $3\ell^2(4\ell + 9)$ elementary steps, each of them involving $O(1)$ additions, comparisons and Euclidean divisions on $\ell + 1$–bit integers.*

Note that extensive experiments show that the average number of iterations is around one. One may give a heuristic explanation to this phenomena [5]. Note also that the algorithm can be generalized to higher dimensions [5].

Bibliography

[1] Avnaim (F.), Boissonnat (J.-D.), Devilliers (O.), F. (Preparata), and Yvinec (M.). – *Evaluating signs of determinants using single-precision arithmetics.* – Research Report n° 2306, Institut National de Recherche en Informatique et en Automatique, 1994.

[2] Daudé (Hervé), Flajolet (Philippe), and Vallée (Brigitte). – An analysis of the Gaussian algorithm for lattice reduction. In Adleman (L.) (editor), *Algorithmic Number Theory Symposium, Lecture Notes in Computer Science*, pp. 144–158. – 1994. Proceedings of *ANTS'94*.

[3] Dupré (A.). – Sur le nombre de divisions à effectuer pour obtenir le plus grand commun diviseur entre deux nombres entiers. *Journal de Mathématiques*, vol. 11, 1846, pp. 41–64.

[4] Lamé (G.). – Note sur la limite du nombre de divisions dans la recherche du plus grand commun diviseur entre deux nombres entiers. *Comptes-Rendus de l'Académie des Sciences*, vol. XIX, 1845, pp. 867–870.
[5] Vallée (B.). – *Algorithme probabiliste pour l'évaluation du signe d'un déterminant $n \times n$ en quasi-simple précision.* – Research report, GREYC, Université de Caen, 1995.
[6] Vallée (B.). – *Evaluation du signe d'un déterminant 2×2: analyse en moyenne.* – Research report, GREYC, Université de Caen, 1995.

Polynomial Solutions of Linear Operator Equations

Marko Petkovšek
University of Ljubljana (Slovenia)

June 7, 1994

[summary by Frédéric Chyzak]

Abstract

The algorithm described here extends the algorithm to find all polynomial solutions of differential and difference equations that was given in [1, 2] to more general operators. It also takes a more efficient approach that avoids using undetermined coefficients. This summary is based on [4].

Let K be a field of characteristic 0 and $L : K[x] \to K[x]$ a K-linear endomorphism of $K[x]$. A new algorithm is presented in [4] that finds all polynomial solutions of homogeneous equations of the form $Ly = 0$, of nonhomogeneous equations of the form $Ly = f$ and of parametric nonhomogeneous equations of the form $Ly = \sum_{i=1}^{m} \lambda_i f_i$. The endomorphisms L under consideration in the following are polynomials in one of the following operators, and with coefficients in $K[x]$:

- the differential operator D defined by $Df(x) = df/dx$;
- the difference operator Δ defined by $\Delta f(x) = f(x+1) - f(x)$;
- the q-dilation operator Q used for q-difference equations and defined by $Qf(x) = f(qx)$. (In this case, $q \in K$, is not zero and not a root of unity.)

The interest of the new algorithm is twofold. First, numerous algorithms need to solve homogeneous, nonhomogeneous or parametric nonhomogeneous equations in $K[x]$ as subproblems. Examples are algorithms to find all rational, hyperexponential, geometric or Liouvillian solutions, to perform indefinite or definite hypergeometric summation, to factorize linear operators, etc. (See for instance [5, 3, 7, 6].) Second, the algorithm that is described here has lower complexity than the usual algorithms, that are often based on undetermined coefficients. The approach here is to find a degree bound on the solutions to be computed, and next find recurrences to compute the coefficients of the solutions efficiently. The problem with undetermined coefficients arises with very concise equations having high degree solutions. Although the number of coefficients to be determined is high, the recurrences that are found by the new algorithm in [4] are of small order.

The idea is to view the space $K[x]$ as a subspace of a unusual space of formal power series, and to embed the space of polynomial solutions into a space of formal power series solutions.

1. Algebraic setup

Let $(P_n(x))_{n\in\mathbb{N}}$ be a sequence of polynomials satisfying the following conditions:

(H1) $\deg P_n = n$, which makes $(P_n(x))_{n\in\mathbb{N}}$ a basis of $K[x]$;
(H2) $P_n \mid P_m$ as soon as $n < m$;

(H3) there exists $(A, B) \in \mathbb{Z}^2$, $A \leq 0$, $A \leq B$, and polynomials $\alpha_i \in K[n]$ such that for all n

$$LP_n = \sum_{i=A}^{B} \alpha_i(n) P_{n+i},$$

with α_A and α_B two non-zero polynomials, and $P_n = 0$ when $n < 0$.

Let $(l_n)_{n\in\mathbb{N}}$ be the dual basis of the $K[x]$-basis $(P_n(x))_{n\in\mathbb{N}}$. By definition, $l_n(P_m) = \delta_{n,m}$ and

$$P_n P_m = \sum_{k\in\mathbb{N}} l_k(P_n P_m) P_k.$$

As a consequence of (H2), computing modulo P_n and considering degrees yields $l_k(P_n P_m) = 0$ when $k < n$. Similarly, $l_k(P_n P_m) = 0$ when $k < m$. It follows that

$$l_k(P_n P_m) = 0 \qquad \text{when} \qquad k < \max(n, m) \text{ or } n + m < k.$$

The next step is to consider formal power series: let $S(x) = \sum_{n=0}^{\infty} c_n P_n(x)$ denote a formal series, and $K[[(P_n(x))_{n\in\mathbb{N}}]]$ the vector space of all such series. Let λ_n be the linear forms over $(P_n(x))_{n\in\mathbb{N}}$ such that $\lambda_n(S) = c_n$. Then $K[[(P_n(x))_{n\in\mathbb{N}}]]$ is a K-algebra for termwise sum and outer product, and for the following inner product

$$ST = \sum_{k=0}^{\infty} \left(\sum_{n,m \leq k \leq n+m} \lambda_n(S)\, \lambda_m(T)\, l_k(P_n P_m) \right) P_k.$$

Thanks to (H3), the operator L is extended to Λ on $K[[(P_n(x))_{n\in\mathbb{N}}]]$ by the following rule:

$$\Lambda S = \sum_{n=0}^{\infty} \left(\sum_{i=-B}^{-A} \alpha_{-i}(n+i)\, \lambda_{n+i}(S) \right) P_n.$$

Now, for any given $f \in K[[(P_n(x))_{n\in\mathbb{N}}]]$, $Ly = f$ is equivalent to

$$\sum_{i=-B}^{-A} \alpha_{-i}(n+i)\, \lambda_{n+i}(y) = l_n(f) \tag{1}$$

for all $n \in \mathbb{N}$ (with $\lambda_k = 0$ when $k < 0$).

The degree bound and the algorithm follow from the following theorems (see [4]).

THEOREM 1. *Let $Ly = f$ where y and f are polynomials. Then*

$$\deg y \leq N = \max\{-B - 1, \deg f - B, n \in \mathbb{Z} \text{ such that } \alpha_B(n) = 0\}.$$

THEOREM 2. *Let N be defined as in the previous theorem. Assume $N \geq 0$. Then, for any formal power series $y \in K[[(P_n(x))_{n\in\mathbb{N}}]]$, the following are equivalent:*

(1) *the formal power series y is a polynomial solution of $Ly = f$;*
(2) *the $\lambda_n(y)$'s satisfy equation (1) for $n \leq N + B$ and $\lambda_n(y) = 0$ for $n < N$.*

2. The algorithm

The goal is to find a basis for the affine space of vectors $g \in K^{N-A+B+1}$ that satisfy

$$\sum_{i=-B}^{-A} \alpha_{-i}(n+i)\, g_{n+i}(y) = l_n(f). \tag{2}$$

Of course, the direction of the affine solution space is given by vectors v that cancel the left hand-side. The vectors g and v represent polynomials in $K[x]$ denoted by

$$h(g,f) = \sum_{n=0}^{N-A+B} g_n P_n \qquad \text{and} \qquad s(v) = \sum_{n=0}^{N-A+B} v_n P_n \qquad \text{respectively.}$$

The set of singularities $\mathcal{S} = \{n \in \mathbb{N} \mid \alpha_A(n) = 0\}$ and the set $\mathcal{N} = \{0, \ldots, A-1\} \cup \mathcal{S}$ play an important rôle. The algorithm proceeds by iteratively computing the coefficients of the solutions. Each time a coefficient is not fully determined by equation (2) or its left hand-side, i.e. for each integer in $\mathcal{N}$, a new parameter is added, along with an equation to guarantee consistency between the elements of the basis under construction.

The algorithm maintains a list of vectors $\mathcal{V}$, a list of indeterminates $\mathcal{I}$, a list of equations $\mathcal{E}$ and an additional vector g. The vectors in $\mathcal{V}$ almost form a basis of the direction space of the affine solution set for the homogeneous equation. Indeed, the solutions are linear combinations of them ruled by the equations in $\mathcal{E}$. The vector g takes the nonhomogeneous part of the equation into account. The algorithm is the following:

(1) set $\mathcal{V}$, $\mathcal{I}$, $\mathcal{E}$ and g to empty lists or vectors;
(2) for each n from 0 to $N-A+B$, perform Step 3 when $n \notin \mathcal{N}$, or Step 4 when $n \in \mathcal{N}$, then go to Step 5;
(3) (*extension step*) extend all vectors $v \in \mathcal{V}$ (resp. g) by using the appropriate instance of equation (2) or its left hand-side;
(4) (*singularity step*) extend all vector $v \in \mathcal{V}$ and g by 0, then add the vector $[0, \ldots, 0, 1]$ (of length $n+1$) to $\mathcal{V}$, add the indeterminate c_n to the list $\mathcal{I}$, and finally add equation (2) for the index $n-A$ (when non-negative), where each g_t has been replaced by the sum cv_t over all pairs $(c, v) \in (\mathcal{I}, \mathcal{V})$ that have been added in a previous singularity step;
(5) let (c_k, v_k) be the pairs added into $(\mathcal{I}, \mathcal{V})$ during singularity steps and $\mathcal{E}(f)$ denote the final list of equations computed by the previous process; perform the following action, according to the type of equation being solved:
   - *homogeneous*: solve the system in the c_k's composed of the equations $\mathcal{E}(f)$ and the equations $\sum_k c_k l_n(s(v_k)) = 0$, for $N < n \leq N-A+B$, and return the general polynomial solution $y = \sum_k c_k s(v_k)$;
   - *nonhomogeneous*: solve the system in the c_k's composed of the equations $\mathcal{E}(f)$ and the equations $\sum_k c_k l_n(s(v_k)) = -l_n(h(g,f))$, for $N < n \leq N-A+B$, and return the general polynomial solution $y = \sum_k c_k s(v_k) + h(g,f)$;
   - *parametric nonhomogeneous*: solve the system in the c_k's and λ_i's composed of the equations $\mathcal{E}(\sum_{i=1}^m \lambda_i f_i)$ and the equations $\sum_k c_k l_n(s(v_k)) = -l_n\left(h\left(g, \sum_{i=1}^m \lambda_i f_i\right)\right)$, for $N < n \leq N-A+B$, and return the general polynomial solution $y = \sum_k c_k s(v_k) + h\left(g, \sum_{i=1}^m \lambda_i f_i\right)$.

3. Choice of the basis $(P_n(x))_{n\in\mathbb{N}}$

The previous algorithm ends by solving a linear system that consists of at most $\sigma - A + B$ equations in at most $\sigma - A$ (resp. $\sigma - A + m$ in the parametric case) variables, where σ is the number of

singularities between $-A$ and $N - A + B$. Therefore, avoiding singularities lessens the complexity. Let $L = \sum_{k=0}^{r} p_k(x)\,\partial^r$, where ∂ is either of D, Δ or Q, and let $d = \max\{j \mid \exists k \quad l_j(p_k) \neq 0\}$. The following choices for the P_n's satisfy conditions (H1–H3).

- *differential case*: $P_n = (x-a)^n/n!$, for a such that $p_r(a) \neq 0$; with this basis, no singularity can occur, $A = r$, $B = d$, and
$$\alpha_i(n) = \sum_{j=0}^{d} \binom{n+i}{j} l_j(p_{j-i});$$
- *difference case*: $P_n = \binom{x-a}{n}$, for $a > \max\{n \in \mathbb{N} \mid p_r(n) = 0\}$; with this basis, no singularity can occur, $A = r$, $B = d$, and
$$\alpha_i(n) = \sum_{k=0}^{r} \sum_{j=0}^{d} \binom{n+i}{j} \binom{j}{i+k} l_j(p_k);$$
- *q-difference case*: $P_n - x^n$; with this basis, singularity can occur, but $A = 0$, $B = d$, and
$$\alpha_i(n) = \sum_{k=0}^{r} q^{nk} l_i(p_k).$$

4. Formal power series solutions

In conclusion, it should be noted that the algorithm that has been discussed can also be used to compute (a description of) formal power series solutions in $(P_n(x))_{n\in\mathbb{N}}$. Indeed, let $M = \max(\mathcal{S})$. Then running the loop of the algorithm for $n = 0, \ldots, M$ and computing the $s(v)$ for $v \in \mathcal{V}$ yields the set of all polynomials $p \in K[x]$ of degree less than or equal to M such that $Lp(x) = f$ modulo $P_{M+1}(x)$. Iterating infinitely many times the extension step (by using equation (2)) yields a formal power series that is a solution.

Bibliography

[1] Abramov (S. A.). – Problems in computer algebra that are connected with a search for polynomial solutions of linear differential and difference equations. *Moscow University Comput. Math. Cybernet.*, n° 3, 1989, pp. 63–68. – Transl. from Vestn. Moskov. univ. Ser. XV Vychisl. mat. kibernet. 3, 56–60.

[2] Abramov (S. A.). – Rational solutions of linear differential and difference equations with polynomial coefficients. *USSR Computational Mathematics and Mathematical Physics*, vol. 29, n° 11, 1989, pp. 1611–1620. – Translation of the Zhurnal vychislitel'noi matematiki i matematichesckoi fiziki.

[3] Abramov (S. A.) and Petkovšek (M.). – Finding all q-hypergeometric solutions of q-difference equations. In Leclerc (B.) and Thibon (J. Y.) (editors), *Formal power series and algebraic combinatorics*. pp. 1–10. – Université de Marne-la-Vallée, 1995. Proceedings SFCA'95.

[4] Abramov (Sergei A.), Bronstein (Manuel), and Petkovšek (Marko). – *On Polynomial Solutions of Linear Operator Equations*. – Technical Report n° 33-468, Institute of Mathematics, Physics and Mechanics, University of Ljubljana. Slovenia., April 1995. 16 pages.

[5] Bronstein (M.) and Petkovšek (M.). – On Ore rings, linear operators and factorisation. *Programmirovanie*, n° 1, 1994, pp. 27–44. – Also available as Research Report 200, Informatik, ETH Zürich.

[6] Gosper (R. William). – Decision procedure for indefinite hypergeometric summation. *Proceedings of the National Academy of Sciences USA*, vol. 75, n° 1, January 1978, pp. 40–42.

[7] Singer (Michael F.). – Liouvillian solutions of linear differential equations with Liouvillian coefficients. *Journal of Symbolic Computation*, vol. 11, 1991, pp. 251–273.

Symbolic and Numerical Manipulations of Divergent Power Series

Jean Thomann
IN2P3, Strasbourg

December 12, 1994

[summary by Bruno Salvy]

Abstract

Divergent series arise naturally in many different contexts. This talk describes mixed symbolic-numerical algorithms to deal with these series when they arise from linear differential equations.

Introduction

A simple example of a divergent numerical series is obtained when summing a Taylor series outside its circle of convergence. More violent divergence is encountered when solving a linear differential equation like the Euler equation

$$x^2y' + y = x$$

by an undeterminate coefficient method. The power series one obtains is the Euler series

$$\sum_{n\geq 0} (-1)^n n! x^{n+1},$$

which has a radius of convergence equal to 0. This problem also occurs in non-linear differential equations, singular perturbations, difference equations, or asymptotic analysis (e.g., by the Laplace method).

The Borel-Ritt theorem states that any power series on any sector of finite opening in the complex plane is the asymptotic expansion of a function which is analytic in the sector. However, this analytic function is far from being uniquely determined, which makes numerical evaluation hopeless. In the context of differential equations the situation is much better because of the following result.

THEOREM 1. *Let $G(x, y_0, \ldots, y_n)$ be an analytic function of $n+2$ variables and $\hat{f} \in \mathbb{C}[[x]]$ a formal power series solution of $G(x, y, \ldots, y^{(n)}) = 0$. Then there exists a real number $k > 0$ such that for all open sectors V with vertex at the origin, opening $< \pi/k$ and small enough radius, there exists a function f which is a solution of the differential equation $G(x, y, \ldots, y^{(n)}) = 0$ asymptotic to $\hat{f}$ on V.*

Thus the main numerical problem is to devise techniques that will sum the divergent series not to values of any analytic function asymptotic to it, but to values of the actual solution of the differential equation corresponding to it.

1. Elementary methods

To compute the sum of a convergent power series $\sum_n a_n x^n$ outside its circle of convergence, one first has to define a path connecting the origin to the point where the sum is desired. A basic subproblem is that of summing along a ray originating at the origin and avoiding singularities of the function. Lindelöf gave a simple way of doing this by computing this value as the limit of

$$a_0 + \lim_{t \to 0} \sum_{n \geq 1} a_n x^n e^{-tn \log n}.$$

When the series is convergent at x, the result is the sum of the series. This technique was generalized by Hardy to sum divergent series of the same type as the Euler series, by computing

$$a_0 + a_1 x + a_2 x^2 + \lim_{t \to 0} \sum_{n \geq 3} a_n x^n e^{-t \log n \log\log n}. \tag{1}$$

Unfortunately, this technique does not behave very well numerically.

The simplest efficient technique to deal with divergent series of the type of the Euler series is called *summation to the least term*. For instance, the values of the successive terms of the Euler series at $x = 1/10$ are

$$.100000, -.010000, .002000, -.000600, .000240, -.000120, .000072, -.000050, .000040, -.000036,$$
$$.000036, -.000040, .000048, -.000062, .000087, -.000131, .000209, -.000356, .000640, -.001216.$$

The absolute value of the terms first decrease, then reaches a minimum at 0.000036, and eventually increase to infinity. By summing these terms up to the smallest one, one gets the numerical value 0.09154563200, which is very close to the value of the corresponding *function* solution of the differential equation, namely 0.09156333394. Using a convergent integral representation for the Euler series, it is not difficult to show (see [5]) that the error made by truncating this series at its least term is exponentially small (with respect to $1/x$). This property is actually much more general (see below). The drawback of this good precision is the impossibility of obtaining an arbitrary precision by this method. This is to be contrasted with the direct summation of convergent power series, where the terms generally first increase before decreasing to 0, but numerous terms are necessary to obtain a good precision. As a consequence, many techniques to convert from various representations of a function to a divergent series have been developed [3].

2. Gevrey asymptotics, Borel transform, k-summability

A good framework to account for the nice behaviour of common divergent series is provided by *Gevrey asymptotics*. A *Gevrey series* is a power series whose coefficients' growth is bounded by $C(n!)^{1/k}A^n$, for some fixed $C, A, k > 0$. Gevrey asymptotic expansions are Gevrey series for which the remainder term satisfies the same type of bound. More precisely, we have the following.

DEFINITION 1. Let k be a positive real number and let V be an open sector with vertex 0. Let f be an analytic function on V. The formal power series $\hat{f} = \sum_{n \geq 0} a_n x^n$ is *Gevrey asymptotic to f of order $s = 1/k$ on V* if for all compact sub-sectors W of V and for all $n \in \mathbb{N}$, there exist $C_W > 0$ and $A_W > 0$ such that

$$|x|^{-n} \left| f(x) - \sum_{p=0}^{n-1} a_p x^p \right| < C_W (n!)^{1/k} A_W^n, \qquad \forall x \in W, \quad x \neq 0$$

By Stirling's formula, truncating a Gevrey asymptotic expansion of order $1/k$ to the least term gives an exponentially small error (in $1/x^k$). This is one of the facets of the interest of Gevrey asymptotics. Another crucial property, due to Watson, is that there is at most one analytic function f Gevrey asymptotic of order $1/k$ to a series $\hat{f}$ on a sector of opening larger than π/k. This provides the uniqueness necessary for numerical computations based on the series alone.

With the same hypotheses as in Theorem 1, an old theorem of Maillet states that there exists $k > 0$ such that $\hat{f}$ is a Gevrey series of order $1/k$. This result is useful in conjunction with a counterpart of Theorem 1 due to Ramis and Sibuya which states that if $\hat{f}$ is Gevrey of order $1/k$ then there exists $k' \geq k$ such that for any sector V with vertex 0, opening $< \pi/k'$ and sufficiently small radius, there exists a function f solution to the differential equation which is Gevrey asymptotic of order k to $\hat{f}$. Combining these two results explains why summing to the least term is a good method for formal series solutions of differential equations.

If $\hat{f} = \sum a_n x^n$ is a Gevrey series of order $1/k$, then its *Borel transform of index k* is defined as

$$(\hat{\mathcal{B}}_k \hat{f})(\xi) = \sum_{n \geq 0} \frac{a_n}{\Gamma(1 + n/k)} \xi^n.$$

For $k = 1$, this corresponds to dividing the n-th coefficient by $n!$. Estimates on the coefficients show that this transform is an analytic function $\phi(\xi)$. Then if the Laplace transform of index k

$$f(x) = \int_d k\phi(\xi) e^{-\xi^k/x^k} \xi^{k-1} d\xi$$

converges, it is called the sum of $\hat{f}$ in the direction d, where d is a straight line from 0 to infinity. The series $\hat{f}$ is then said to be *k-summable in the direction d*. The convergence of this integral is related to the growth of ϕ at infinity. It is easy to see that the Taylor series of f is precisely $\hat{f}$ so that this process yields a convergent representation for $\hat{f}$. The sum however depends on the path of integration d, in the same way an analytic continuation depends on a path. This dependency is related to the *Stokes phenomenon*.

Numerically, in the case of convergence, the problem is reduced to finding k and computing the analytic continuation of ϕ. In the case of solutions of *linear* differential equations, this computation is simplified by noticing that k can be deduced from the slopes of a Newton polygon associated with the linear differential equation and that ϕ satisfies a linear differential equation derived formally from that satisfied by $\hat{f}$. Therefore its Taylor coefficients satisfy a computable linear recurrence which can be used to obtain many coefficients efficiently. Besides, the possible singularities of ϕ are located at the zeroes of the leading coefficient of the linear differential equation it satisfies, so that it is possible to compute the continuation along a path which avoids singularities, with a knowledge of the exact radius of convergence of the power series one is computing. This process can also be applied to the divergent series that occur as part of the asymptotic expansion of solutions of linear differential equations at an irregular singular point, by first computing a linear differential equation satisfied by these series.

3. Multisummability

Not all solutions of linear differential equations are k-summable for some k. One reason for this is that the order of growth of an analytic function at infinity is related to the growth of its Taylor coefficients at the origin. Thus by adding a 1-summable and a 2-summable divergent series, one obtains a series which is Gevrey of order 1, but the growth at infinity of its Borel transform of level 1 is exponential of order 2. This leads to the consideration of a more general class of divergent series.

DEFINITION 2. Let $k_1, \ldots, k_r$ be real numbers such that $k_1 > \cdots > k_r > 0$ and let d be a line from 0 to infinity. A formal power series $\hat{f}(x)$ is $(k_1, \ldots, k_r)$-*summable in the direction* d if there exists a positive integer m such that $\hat{f}(x^{1/m})$ is a sum of r series $\hat{f}_1, \ldots, \hat{f}_r$, each $\hat{f}_i$ being k_i/m summable in the direction d.

A result of Jurkat is that Hardy's summation technique (1) will sum any multisummable series without having to know $k_1, \ldots, k_r$.

The following recent theorem due to Braaksma demonstrates the relevance of multisummability.

THEOREM 2. *Let $G(x, y_0, \ldots, y_n)$ be an analytic function of $n+2$ variables and $\hat{f} \in \mathbb{C}[[x]]$ a formal power series solution of $G(x, y, \ldots, y^{(n)}) = 0$. Let $k_1 > \cdots > k_r > 0$ be the positive slopes of the associated Newton polygon. Then $\hat{f}$ is $(k_1, \ldots, k_r)$ summable in every direction d, except possibly a finite number of them.*

Braaksma's proof uses Écalle's theory of accelero-summability.

In the linear case, a technique due to Balser makes it possible to compute the sum by computing successive Borel transforms of indices κ_j related to the k_i's by $1/k_j = 1/\kappa_1 + \cdots + 1/\kappa_j$ and then recovering the function by computing the corresponding Laplace transforms of order κ_j in reverse order. At each step, exact linear differential equations can be computed for the various Taylor series and exact linear recurrences for their coefficients.

Conclusion

Numerically, the difficulty is that each level of integration is time consuming and induces a precision loss. At the moment, this process is still largely interactive, notably the choice of paths of integration at each step.

Bibliography

[1] Balser (Werner). – *From Divergent Power Series to Analytic Functions.* – Springer-Verlag, 1994, *Lecture Notes in Mathematics*, vol. 1582.

[2] Braaksma (B. L. J.). – Multisummability of formal power series solutions of nonlinear meromorphic differential equations. *Annales de l'Institut Fourier*, vol. 42, n° 3, 1992, pp. 517–540.

[3] Dingle (Robert B.). – *Asymptotic Expansions: Their Derivation and Interpretation.* – Academic Press, London, New York, 1973.

[4] Hardy (G. H.). – *Divergent Series.* – Oxford University Press, 1947.

[5] Olver (F. W. J.). – *Asymptotics and Special Functions.* – Academic Press, 1974.

[6] Ramis (J.-P.) and Sibuya (Y.). – Hukuhara domains and fundamental existence and uniqueness theorems for asymptotic solutions of Gevrey type. *Asymptotic Analysis*, vol. 2, 1989, pp. 39–94.

[7] Ramis (J.-P.) and Sibuya (Y.). – A new proof of multisummatility of formal solutions of non linear meromorphic differential equations. *Annales de l'Institut Fourier*, vol. 44, n° 3, 1994, pp. 811–848.

[8] Ramis (Jean-Pierre). – *Séries divergentes et théories asymptotiques.* – Société Mathématique de France, 1993, *Panoramas et Synthèses*, vol. 121.

[9] Thomann (Jean). – Resommation des séries formelles. Solutions d'équations différentielles linéaires du second ordre dans le champ complexe au voisinage de singularités irrégulières. *Numerische Mathematik*, vol. 58, n° 5, 1990, pp. 503–535.

[10] Thomann (Jean). – Procédés formels et numériques de sommation de séries solutions d'équations différentielles. – Preprint, April 1994.

Holonomic Systems and Automatic Proofs of Identities

Frédéric Chyzak
INRIA and École Polytechnique

October 3, 1994

[summary by Bruno Salvy]

Abstract

D. Zeilberger has shown how many combinatorial identities involving special functions can be proved using the theory of holonomic sequences and functions. This work presents a general algorithmic approach to the multivariate case, together with an implementation.

Introduction

Speaking informally, D. Zeilberger has defined holonomic functions in [10] as those functions of one or several variables satisfying sufficiently many *linear* equations (differential equations or recurrence relations) with polynomial coefficients so that they are completely determined by a finite number of initial conditions and a finite number of polynomial coefficients. The study of these functions is motivated by their pervasiveness in combinatorics and special functions theory. The class of holonomic functions enjoys closure properties that make it possible to construct equations satisfied by a particular function from equations satisfied by simpler functions. These operators can be exploited by many algorithms. In particular, series expansions of holonomic function can be computed efficiently and asymptotic estimates related to them can be derived from the operators. A very important special class of holonomic functions is formed by algebraic functions, for which finding a differential operator is the best known algorithm to compute series expansions of large order.

For the single variable case, the *gfun* package [6] provides functions that construct recurrence or differential operators satisfied by holonomic sequences or functions and thus prove formulæ. For instance, Cassini's identity on the Fibonacci numbers

$$F_{n+2}F_n - F_{n+1}^2 = (-1)^n,$$

is proved by computing a linear recurrence satisfied by the left-hand side, starting from the linear recurrence satisfied by the Fibonacci numbers. Here is the kind of proof for which *gfun* provides tools:

$$\begin{aligned} h_n &= F_{n+2}F_n - F_{n+1}^2 = F_n^2 + F_nF_{n+1} - F_{n+1}^2 \\ h_{n+1} &= F_{n+1}^2 + F_{n+1}F_{n+2} - F_{n+2}^2 = F_{n+1}^2 - F_{n+1}F_n - F_n^2 = -h_n. \end{aligned}$$

It is then sufficient to check that $(-1)^n$ also satisfies this recurrence and that a finite number of initial conditions match.

Ore operator	$\sigma(x)$	$\delta(x)$	Action
differentiation	x	1	$f(x) \mapsto f'(x)$
shift	$x+1$	0	$f(x) \mapsto f(x+1)$
difference	$x+1$	1	$f(x) \mapsto f(x+1) - f(x)$
q-dilation	qx	0	$f(x) \mapsto f(qx)$
q-differentiation	qx	1	$f(x) \mapsto [f(qx) - f(x)]/[(q-1)x]$
Mahlerian operator	x^p	0	$f(x) \mapsto f(x^p)$

TABLE 1. Examples of Ore operators

In one variable, the algorithms for differential equations and recurrence equations are always very similar and can be profitably expressed using the vocabulary of *Ore operators* [5]. These are defined over a field $K(x)$ by a commutation rule

$$\partial x = \sigma(x)\partial + \delta(x), \tag{1}$$

where σ is a ring endomorphism of $K(x)$ and δ is a vector-space endomorphism of $K(x)$. Table 1 gives a list of important examples.

In several variables, a holonomic function is defined by several operators and most of the closure properties still hold. In addition, holonomy is often preserved by specialization; by definite and indefinite summation (for recurrences) or integration (for differential equations). However, making the corresponding construction of operators explicit is more difficult. Wilf and Zeilberger [9] have given efficient algorithms for some of these operations in the *hypergeometric* and the *q-hypergeometric* case (linear recurrences or q-recurrences of order 1 on a sequence or on the coefficients of a series). N. Takayama [7, 8] has used Gröbner bases of differential, difference and q-difference operators to make an explicit construction of operators in the general (non-hypergeometric) case. This is (at least partly) implemented in his programs *Kan* and *Macaulay for D-modules*.

The aim of this work is to attack multivariate holonomy via Ore operators and non-commutative elimination by Gröbner bases or a skew version of the Euclidean algorithm [1, 2]. This is implemented in F. Chyzak's *Mgfun* package[1] written in Maple.

1. Elimination and ideals

Ore algebras in the univariate case are algebras $K\langle x, \partial\rangle$ where K is a ring and x and ∂ are related by (1). The multivariate case is obtained as the tensor product $K\langle x_1, \partial_1\rangle \otimes \cdots \otimes K\langle x_n, \partial_n\rangle$.

The example of Legendre polynomials illustrates a simple use of elimination. Legendre polynomials satisfy the following three dependent relations:

$$(1-x^2)P_n''(x) - 2xP_n'(x) + n(n+1)P_n(x) = 0, \tag{2}$$

$$(n+2)P_{n+2}(x) - (2n+3)xP_{n+1}(x) + (n+1)P_n(x) = 0, \tag{3}$$

$$(1-x^2)P_{n+1}'(x) + (n+1)xP_{n+1}(x) - (n+1)P_n(x) = 0. \tag{4}$$

Any of these relations can be deduced from the other ones. Here is how *Mgfun* can be used to prove (3) from (2) and (4). The computation consists in defining a suitable Ore algebra, a proper term ordering on the variables, and then computing a Gröbner basis with respect to this order.

[1]Available by anonymous ftp on `ftp.inria.fr:INRIA/Projects/algo/programs/Mgfun` or at the URL `http://www-rocq.inria.fr/Combinatorics-Library/www/programs/Mgfun`.

```
A:=orealg([x,diff,Dx],[n,shift,Sn]): T:=termorder(A,plex=[Dx,Sn],max):
DE:=(1-x^2)*Dx^2-2*x*Dx+n*(n+1): RDE:=(1-x^2)*Dx*Sn+(n+1)*x*Sn-(n+1):
map(collect,gbasis([DE,RDE],T,ratpoly(rational,[n,x])),Sn,factor);
```

$$[(-n-1)S_n + xn - D_x + x^2 D_x + x, \quad (-n-2)S_n^2 + x(2n+3)S_n - n - 1]$$

The operator (3) appears at the end of the basis. The same result can be obtained by a skew Euclidean algorithm applied to (2) and (4). This is done in *Mgfun* as follows:

```
RE:=skewelim(DE,RDE,Dx,A,ratpoly(rational,[n,x])):
```

Since we are interested in functions or sequences annihilated by operators, it is natural to consider the left ideal generated by these operators. In one variable, the ring $K(x)\langle\partial\rangle$ is Euclidean (therefore principal) [5]. Thus one can work with solutions of univariate Ore operators as one works with algebraic numbers, using Euclid's algorithm to compute normal forms in a finite dimensional vector space generated by $1, \partial, \partial^2, \ldots, \mathcal{P}$ where $\mathcal{P}$ is an operator analogous to the minimal polynomial. These normal forms in turn are used to compute operators annihilating sums or products of holonomic functions by performing computations in the proper finite dimensional vector space and determining a linear relation by Gaussian elimination. In several variables, skew polynomial rings are Noetherian [3] so that a normal form is provided by Gröbner bases. The same kind of algorithms as in the univariate case apply. Elimination between operators consists in finding an element of the ideal they generate which does not contain the undesirable variable. A recursive extended gcd algorithm can be used to eliminate a ∂ variable between two operators. The general case of elimination is obtained by Gröbner bases with appropriate orders.

2. Creative telescoping

Holonomic ideals form an important class of ideals of operators. In these ideals, it is possible to eliminate *any* of the variables. This elimination is applied by *creative telescoping* [11] to the computation of definite integrals or sums. The idea is that if f is annihilated by a holonomic ideal of $K\langle x, \partial\rangle$ and if the $\partial^k f$'s $(k \geq 0)$ vanish at the border $\partial\Omega$ of a suitable domain Ω, then an operator annihilating $\partial_\Omega^{-1} f$ (the definite sum or integral) is obtained by first eliminating x. This yields an operator which can be rewritten $\partial A(\partial) + B$ such that

$$[\partial \cdot A(\partial)](f) + B(f) = 0.$$

Then applying ∂_Ω^{-1} (i.e., summing or integrating over the domain) gives $A(\partial)(f)|_{\partial\Omega} + \partial_\Omega^{-1} B(f) = 0$, where the hypotheses ensures that the first part (the sum or integral at the boundary) is zero. Since B does not contain x it commutes with ∂^{-1} and is the desired operator.

As an example, we compute a system of differential equations satisfied by the generating function of the Legendre polynomials

$$F(x,z) = \sum_{n \geq 0} P_n(x) z^n$$

starting from (2) and (3). The steps to be performed are: *(i)* creation of the Ore algebra $\mathcal{A} = \mathbb{Q}\langle n, S_n\rangle \otimes \mathbb{Q}\langle x, \partial_x\rangle \otimes \mathbb{Q}\langle z, \partial_z\rangle$; *(ii)* determination of operators annihilating $P_n(x)z^n$ in $\mathcal{A}$; *(iii)* elimination of n; *(iv)* left division by $S_n - 1$. Here is the corresponding *Mgfun* session:

```
Legendre:=[RE,DE,Dz]: z_to_the_n:=[Dx,z*Dz-n,Sn-z]:
A:=orealg([x,diff,Dx],[n,shift,Sn],[z,diff,Dz]):
T1:=termorder(A,tdeg=[Dx,Sn,Dz],max):
Legendre_times_zn:=hprod(Legendre,z_to_the_n,2,T1);
```

Legendre_times_zn := $[D_x^2 - x^2 D_x^2 - 2xD_x + n^2 + n, nS_n^2 + 2Sn^2 + z^2 n + z^2 - 2zxnS_n - 3zxS_n, zD_z - n]$

```
T2:=termorder(A,lexdeg=[[n],[Dx,Sn,Dz]],max):
gb:=gbasis(Legendre_times_zn,T2,ratpoly(rational,[x,n,z]));
```

$$gb := [D_x^2 - x^2 D_x^2 - 2xD_x + z^2 D_z^2 + 2zD_z, z - xS_n + S_n^2 D_z - 2zxS_n D_z + z^2 D_z, zD_z - n]$$

```
map(collect,subs(Sn=1,[gb[1],gb[2]]),[Dx,Dz]);
```

$$[(1-x^2)D_x^2 - 2xD_x + 2zD_z + z^2 D_z^2, (1 - 2zx + z^2)D_z + z - x]$$

This is a system of differential equations satisfied by the generating function of the Legendre polynomials. The whole computation is performed in 2.1 s. on a Dec Alpha. The system can be further simplified by another elimination to yield an operator of second order in D_x only. From the above system a symbolic solver of differential equations can be used to find the well-known formula

$$\sum_{n \geq 0} P_n(x) z^n = \frac{1}{\sqrt{1 - 2xz + z^2}}.$$

Conclusion

This approach is susceptible to numerous applications, extensions and improvements. Applications to q-computations look promising; Comtet's algorithm [4] to compute the differential equation satisfied by an algebraic function can be generalized to some extent; a program handling operators and initial conditions simultaneously could benefit from the initial conditions to avoid letting the orders of the operators grow too much and thus could turn into an efficient formulæ prover; computation of Gröbner bases could be speeded up using a non-commutative analogue of trace lifting or simple generalizations of the FGLM algorithm, etc. Hopefully, all of this will appear in F. Chyzak's thesis.

Bibliography

[1] Chyzak (Frédéric). – *Holonomic Systems and Automatic Proofs of Identities.* – Research Report n° 2371, Institut National de Recherche en Informatique et en Automatique, October 1994.

[2] Chyzak (Frédéric) and Salvy (Bruno). – Non-commutative elimination in Ore algebras proves multivariate holonomic identities. – In preparation, 1995.

[3] Cohn (P. M.). – *Free Rings and Their Relations.* – Academic Press, 1971, *London Mathematical Society Monographs.*

[4] Comtet (L.). – Calcul pratique des coefficients de Taylor d'une fonction algébrique. *L'Enseignement Mathématique*, vol. 10, 1964, pp. 267–270.

[5] Ore (Oystein). – Theory of non-commutative polynomials. *Annals of Mathematics*, vol. 34, 1933, pp. 480–508.

[6] Salvy (Bruno) and Zimmermann (Paul). – Gfun: a Maple package for the manipulation of generating and holonomic functions in one variable. *ACM Transactions on Mathematical Software*, vol. 20, n° 2, 1994, pp. 163–177.

[7] Takayama (Nobuki). – Gröbner basis, integration and transcendental functions. In *Symbolic and algebraic computation.* ACM, pp. 152–156. – 1990. Proceedings ISSAC'90, Kyoto.

[8] Takayama (Nobuki). – An approach to the zero recognition problem by Buchberger algorithm. *Journal of Symbolic Computation*, vol. 14, 1992, pp. 265–282.

[9] Wilf (Herbert S.) and Zeilberger (Doron). – An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities. *Inventiones Mathematicae*, vol. 108, 1992, pp. 575–633.

[10] Zeilberger (Doron). – A holonomic systems approach to special functions identities. *Journal of Computational and Applied Mathematics*, vol. 32, n° 3, 1990, pp. 321–368.

[11] Zeilberger (Doron). – The method of creative telescoping. *Journal of Symbolic Computation*, vol. 11, 1991, pp. 195–204.

Short and Easy Computer Proofs of Partition and q-Identities

Peter Paule
RISC Linz, Austria

October 3, 1994

[summary by Bruno Salvy]

1. Binomial and hypergeometric identities ($q = 1$)

Binomial identities are identities which involve binomial coefficients like the famous *Saalschütz identity:*

$$\sum_{k=0}^{n} \frac{\binom{n}{k}\binom{x}{k}\binom{y}{k}}{\binom{x+y+z+n}{k}\binom{z+k}{k}} = \frac{\binom{x+z+n}{n}\binom{y+z+n}{n}}{\binom{z+n}{n}\binom{x+y+z+n}{n}}. \tag{1}$$

Tables like [3] list several hundred such identities. Since binomial coefficients satisfy many relations, the expression on the left-hand side may appear under numerous disguises, which makes it difficult to locate it in such tables (or to implement table lookup in a computer algebra system). However, a sort of *normal form* follows from the observation that in many identities with left-hand side $\sum_k f(k)$, the function $f(k)$ satisfies

$$\frac{f(k+1)}{f(k)} \in F(k), \tag{2}$$

for some suitable field of coefficients F. Thus f is completely determined by $f(0)$ and a rational function, for which a normal form is available. A function f satisfying this property is called a *hypergeometric term.* In a suitable algebraic extension, $f(k)$ can be made explicit:

$$f(k) = \frac{(a_1)_k \cdots (a_m)_k}{(b_1)_k \cdots (b_n)_k} \frac{z^k}{k!} f(0),$$

where $(a)_k = a(a+1)\cdots(a+k-1)$ denotes the rising factorial. The sum of f (when $f(0) = 1$) is usually called the *hypergeometric series* with the following notation

$${}_mF_n\left(\left.\begin{matrix} a_1, \ldots, a_m \\ b_1, \ldots, b_n \end{matrix}\right| z\right) = \sum_{k=0}^{\infty} f(k).$$

According to G. E. Andrews, "By using hypergeometric series one can reduce 450 of the 577 entries in Gould's table to 32 entries." Thus for instance, the Saalschütz identity is obtained as

$${}_3F_2\left(\left.\begin{matrix} -x, -y, -n \\ z+1, -x-y-z-n \end{matrix}\right| 1\right) = \frac{(x+z+1)_n (y+z+1)_n}{(z+1)_n (x+y+z+1)_n}.$$

Given a function $F(n,k)$ hypergeometric in both parameters, plus a technical condition (holonomy), D. Zeilberger gave an algorithm to compute a linear recurrence satisfied by the definite sum with respect to one of the parameters. The technique is based on *creative telescoping* [11] which

applies to a larger context of holonomic identities. To compute $\sum_k F_{n,k}$ from a first-order recurrence like (2) in n and a second one in k, the idea is to determine a recurrence satisfied by $F_{n,k}$ where k does not appear *in the coefficients*. In the case of the Saalschütz identity, this gives

$$
\begin{aligned}
&(n+3+z)(n+1+x+y+z)_3 F_{n+3,k+1}\\
&-(n+1+x+y+z)_2\{[(x+y+z+2n+5)(2n+z+5)-2(n+2)(n+3)]F_{n+2,k+1}\\
&\qquad +(n+2-y)(n+2-x)F_{n+2,k}\}\\
&+(n+1+x+y+z)[(n+2)(n+2+x+y+2z)(n+2+x+y+z)F_{n+1,k+1}\\
&\qquad +(n+2)(2n^2+6n+2nz-x^2-xz-yz+3z-y^2+5)F_{n+1,k}]\\
&-(n+1)(n+2)(n+y+z+1)(n+x+z+1)F_{n,k}=0.
\end{aligned}
\tag{3}
$$

In the holonomic universe, such an elimination is always possible. The above identity is then rewritten

$$
\begin{aligned}
&(x+y+z+n+1)_3(n+z+3)F_{n+3,k}\\
&-(x+y+z+n+1)_2\times\\
&\qquad (z^2+xz+10z+yz+4nz+xy+14n+3y+ny+17+nx+3x+3n^2)F_{n+2,k}\\
&+(x+y+z+n+1)(n+2)\times\\
&\qquad (2xz+2xy+2nx+4x+2z^2+9z+5nz+3n^2+9+10n+4y+2yz+2ny)F_{n+1,k}\\
&-(n+2)(n+1)(n+1+x+z)(n+1+y+z)F_{n,k}=G_{n,k+1}-G_{n,k},
\end{aligned}
\tag{4}
$$

with

$$
\begin{aligned}
G_{n,k}=&(x+y+z+n+2)(x+y+z+n+1)[(x+y+z+n+3)(n+z+3)F_{n+3,k}\\
&-(z^2+4nz+10z+xz+2n^2+10n+5x+13+2nx+2ny+yz+5y)F_{n+2,k}\\
&+(n+2)(2z+2+n+x+y)F_{n+1,k}].
\end{aligned}
\tag{5}
$$

Now summing with respect to k shows that the left-hand side of (4) is the desired recurrence for the sum. Using M. Petkovšek's algorithm [7], it is then possible to find the right-hand side of (1).

H. Wilf and D. Zeilberger have designed a *fast* algorithm (as opposed to general non-commutative elimination) to compute recurrences of the type (3) for terminating hypergeometric summation and multi-summation [9]. The analogous of $G_{n,k}$ in (5) is called a *certificate* of the computation, since it makes (4) easy to check by mere rational function manipulations. This algorithm has been at least partially implemented in Maple by D. Zeilberger [10] and T. Koornwinder [4] and in Mathematica by P. Paule and M. Schorn [5]. It has the extra advantage that the recurrence it returns instead of (3) is of order 1.

2. q-identities

A natural generalization of hypergeometric identities is provided by q-hypergeometric identities. In this context, rising factorials are replaced by $(a)_n = (1-a)(1-aq)\cdots(1-aq^{n-1})$, $n!$ becomes $(q)_n/(1-q)^n$ and the binomial coefficients become the q-binomial coefficients or Gaussian polynomials

$$
\begin{bmatrix} n \\ k \end{bmatrix} = \frac{(q)_n}{(q)_k (q)_{n-k}}, \qquad 0 \le k \le n.
$$

The classical counterpart of these numbers or identities involving them is obtained by letting q tend to 1. Like binomial coefficients, the Gaussian polynomials have nice combinatorial interpretations and properties (see [2]).

A q-hypergeometric term is a function $f(k)$ such that $f(k+1)/f(k)$ is a rational function of q and q^k. The techniques of Wilf and Zeilberger extend to q-hypergeometric identities [9] and a Mathematica implementation is available [8]. For instance, the following identity is equivalent to the q-binomial theorem:

$$\sum_{k=-n}^{n} q^{\binom{k}{2}} \begin{bmatrix} 2n \\ n+k \end{bmatrix} x^k = (-x)_n(-q/x)_n. \tag{6}$$

Let the summand be $f_{n,k}$, which satisfies

$$\frac{f_{n,k+1}}{f_{n,k}} = x\frac{q^k - q^n}{1-q^{n+k+1}}, \qquad \frac{f_{n+1,k}}{f_{n,k}} = \frac{(1-q^{2n+1})(1-q^{2n+2})q^k}{(q^k-q^{n+1})(1-q^{n+k+1})}.$$

Then it can be found that $f_{n,k}$ also satisfies the following recurrence where k does not appear in the coefficients:

$$q^{n+1}f_{n,k+2} + x(1+q^{2n+1})f_{n,k+1} + x^2q^n f_{n,k} - x f_{n+1,k+1} = 0.$$

This is then rewritten as

$$x f_{n+1,k} - (x^2q^n + xq^{2n+1} + x + q^{n+1})f_{n,k} = g_{n,k+1} - g_{n,k}, \tag{7}$$

with certificate

$$g_{n,k} = x f_{n+1,k} - q^{n+1}f_{n,k+1} - xq(q^{2n} + q^n + 1)f_{n,k}.$$

From this follows that the left-hand side of (7) gives a recurrence satisfied by the sum. Since this recurrence is of order 1, solving it is easy and this yields the right-hand side of (6).

It is not difficult to see that $\lim_{n\to\infty} \begin{bmatrix} 2n \\ n+k \end{bmatrix} = 1/(q)_\infty$. Now, taking the limit in (6), changing q into q^2 and x into qz yields the famous *Jacobi triple-product identity:*

$$\sum_{k=-\infty}^{\infty} q^{k^2} z^k = \prod_{n=0}^{\infty}(1-q^{2n+2})(1+zq^{2n+1})(1+\frac{q^{2n+1}}{z}). \tag{8}$$

3. Partition identities

There is a strong connection between identities about partitions and q-calculus. For instance the q-binomial coefficient $\begin{bmatrix} N+M \\ M \end{bmatrix}$ is the generating function (in the variable q) of the number of partitions of n into at most M parts, each $\leq N$. Probably the most famous partition identities in this category are the *Rogers-Ramanujan identities*, an example of which is

$$1 + \sum_{k=1}^{\infty} \frac{q^{k^2}}{(1-q)(1-q^2)\cdots(1-q^k)} = \prod_{m=0}^{\infty} \frac{1}{(1-q^{5m+1})(1-q^{5m+4})}. \tag{9}$$

It is easy to see that the right-hand side is the generating function of partitions into parts equal to 1 or 4 mod 5. It turns out that the left-hand side can be read as the generating function of partitions into parts with minimal difference 2 (see [2]). This identity states that these numbers are identical. For instance, the coefficient of q^{10} is 6 on both sides, corresponding to (10), (1,9), (2,8), (3,7) (4,6), (1,3,6) on the left and (1^{10}), $(1^6,4)$, $(1^2,4^2)$, $(1^4,6)$, (1,9), (4,6) on the right (exponent denoting repetition).

D. Zeilberger has used his algorithm to prove (9) by proving the following finite version due to G. Andrews:

$$\sum_k \frac{q^{k^2}}{(q)_k(q)_{n-k}} = \sum_k \frac{(-1)^k q^{(5k^2-k)/2}}{(q)_{n-k}(q)_{n+k}}. \tag{10}$$

Letting n tend to infinity and making use of Jacobi's identity (8) yields (9). However, as in many other instances, the WZ-technique (for Wilf & Zeilberger) yields a recurrence whose order is not minimal (here 5 instead of 2). Not only is it not mathematically aesthetic, but it generally leads to computations that require much more memory and computer time than necessary.

P. Paule's key observation [6] is that *by taking advantage of the symmetry in the summands of (10) (and of many other similar identities) the order of the equation obtained by the WZ-algorithm becomes minimal!* Thus for sums with even summand $f(k)$, the idea is to try summing $(f(k)+f(-k))/2$ instead. This leads to the following three-line proof of (9).

THEOREM 1. *The Rogers-Ramanujan identity*

$$\sum_{k=0}^{\infty} \frac{q^{k^2}}{(q)_k} = \frac{1}{(q)_\infty} \sum_k (-1)^k q^{(5k^2-k)/2}$$

is the limit when $n \to \infty$ of

$$\sum_k \frac{2q^{k^2}}{(q)_k (q)_{n-k}} = \sum_k \frac{(-1)^k q^{(5k^2-k)/2}(1+q^k)}{(q)_{n-k}(q)_{n+k}}. \tag{11}$$

In addition, both sides of (11) satisfy the recurrence

$$(1-q^n)u_n = (1+q-q^n+q^{2n-1})u_{n-1} + qu_{n-2}.$$

PROOF. The initial conditions $u_0 = 2$ and $u_1 = 2(1+q)/(1-q)$ are easily seen to hold for both sides. The proof that both sides satisfy the recurrence relation is easy once given their certificates: $-q^{2n-1}(1-q^{n-k})$ and $q^{2n+3k}(1-q^{n-k})/(1+q^k)$. □

Bibliography

[1] Abramov (S. A.) and Petkovšek (M.). – Finding all q-hypergeometric solutions of q-difference equations. In Leclerc (B.) and Thibon (J. Y.) (editors), *Formal power series and algebraic combinatorics*. pp. 1–10. – Université de Marne-la-Vallée, 1995. Proceedings SFCA'95.

[2] Andrews (George E.). – *The Theory of Partitions*. – Addison-Wesley, Reading, Massachusetts, 1976, *Encyclopedia of Mathematics and its Applications*, vol. 2.

[3] Gould (Henry Wadsworth). – *Combinatorial Identities; a standardized set of tables listing 500 binomial coefficient summations*. – Morgantown, W. Va., 1972, revised edition.

[4] Koornwinder (Tom H.). – On Zeilberger's algorithm and its q-analogue. *Journal of Computational and Applied Mathematics*, vol. 48, 1993, pp. 91–111.

[5] Paule (P.) and Schorn (M.). – A Mathematica version of Zeilberger's algorithm for proving binomial coefficient identities. *Journal of Symbolic Computation*, 1995. – To appear.

[6] Paule (Peter). – Short and easy computer proofs of the Rogers-Ramanujan identities and of identities of similar type. *The Electronic Journal of Combinatorics*, vol. 1, n° 10, 1994, pp. 1–9.

[7] Petkovšek (Marko). – Hypergeometric solutions of linear recurrences with polynomial coefficients. *Journal of Symbolic Computation*, vol. 14, 1992, pp. 243–264.

[8] Riese (A.). – A Mathematica q-analogue of Zeilberger's algorithm for proving q-hypergeometric identities. – Diploma Thesis. In preparation.

[9] Wilf (Herbert S.) and Zeilberger (Doron). – An algorithmic proof theory for hypergeometric (ordinary and "q") multisum/integral identities. *Inventiones Mathematicae*, vol. 108, 1992, pp. 575–633.

[10] Zeilberger (Doron). – A Maple program for proving hypergeometric identities. *SIGSAM Bulletin*, vol. 25, n° 3, July 1991, pp. 4–13.

[11] Zeilberger (Doron). – The method of creative telescoping. *Journal of Symbolic Computation*, vol. 11, 1991, pp. 195–204.

Effective Identity Testing in Extensions of Differential Fields

Ariane Péladan-Germa
GAGE, École polytechnique (France)

November 21, 1994

[summary by Frédéric Chyzak]

Abstract

A. Péladan-Germa deals with extensions of differential rings by solutions of systems of PDE's. In the case of ODE's, the problem of equality testing in the extension ring has been solved [2, 3]. The author gives an algorithm for the more general case of PDE's [6]. It is based on the theory of differential algebra, and in particular on the concept of auto-reduced coherent sets [5, 8].

1. Outline of the algorithm

Let R be the polynomial ring $k[x_1, \ldots, x_n]$ endowed with the usual partial derivatives ∂_{x_i}. The work described here gives an algorithm for effective equality testing in differential extensions of R by series defined by algebraic partial differential equations. More precisely, let $f_i \in k[[x_1, \ldots, x_n]]$ be formal power series defined by equations of the form

$$Q_h\left(x_1, \ldots, x_n, \{\partial_\alpha f_i\}\right) = 0, \tag{1}$$

for polynomials Q_h in finitely many derivatives $\partial_\alpha f_i$. Given these polynomials Q_h and similar polynomials P_h, the problem is to decide whether the f_i's satisfy the equations represented by the P_h's, and in case they do not, to return one of the P_h's that is not satisfied.

The viewpoint adopted here is to consider the formal power series $P_h\left(x_1, \ldots, x_n, \{\partial_\alpha f_i\}\right)$ as elements of the differential extension of R by the f_i's. However, she requires an assumption on these power series, namely that they are defined by a *complete system*. Informally, a complete system provides with sufficiently many equations and initial conditions so as to be able to compute *any* coefficient of *any* of the power series f_i (see Theorem 2 below). The same also applies to all series $P_h\left(x_1, \ldots, x_n, \{\partial_\alpha f_i\}\right)$. The algorithm decides whether *all* coefficients are zero. Moreover, a complete system makes sure that the algorithm will work for any set of P_h's, even for badly conditioned ones. Given the set $\mathcal{A}$ of all P_h's and all Q_h's, the algorithm is:

(1) Compute an *autoreduced coherent set* $\mathcal{B}$ associated to $\mathcal{A}$ and an additional polynomial H, the product of all *initials* and *separants* of the elements of $\mathcal{B}$. (These notions are defined below.) Informally, the set $\mathcal{B}$ defines the same series as $\mathcal{A}$, up to possible singularities described by H: the algorithm has to decide whether $H(f) = 0$.

(2) To this end:
- if $H(f)(0) \neq 0$ (*regular case*), then the $P_h(f)$ are all zero if and only if $\partial_\alpha B(f)(0) = 0$ for a computable finite set of derivatives and all $B \in \mathcal{B}$;
- otherwise, the algorithm is applied recursively to decide whether all $P_h(f)$'s and $H(f)$ are zero; then:

- if $H(f) \neq 0$ (*semi-regular case*), the problem reduces again to testing $\mathcal{B}(f) = 0$; $\mathcal{B}(f)$ continuously depends on the initial conditions defining f, and decision is done by computing a Groebner basis in an usual non-differential algebra to find the closure of an appropriate algebraic variety;
- if one of the P_h's, say P_k, is not cancelled, return the answer $P_k(f) \neq 0$;
- otherwise (*singular case*), return the answer that all $P_h(f)$'s are zero.

Termination of this recursive algorithm is ensured by Theorem 1 below.

2. Differential algebra

A suitable theory to work with equations like (1) is the theory of *differential algebra* [5, 8]. Polynomials like the P_h's and the Q_h's are called *partial differential polynomials*, in short *pdp*'s, and form the *ring of partial differential polynomials* $\mathcal{R} = R[\{\partial_\alpha y_i\}]$. Note that this ring is a commutative ring in infinitely many indeterminates.

Differential algebra theory introduces *differential ideals*, i.e. ideals closed under all differentiations. Usual ideals are called *algebraic* ideals. For given polynomials P_i, the algebraic ideal of $\mathcal{R}$ is denoted $(P_1, \ldots, P_t)$, while the differential ideal is denoted $[P_1, \ldots, P_t]$. In fact, the differential ideal $[P_1, \ldots, P_t]$ is the algebraic ideal generated by all $\partial_\alpha P_i$'s.

The problem of working with (algebraic) ideals in usual non-differential algebras of polynomials is solved by Groebner bases computations. Similar tools have been developed in the differential case: first, a process of reduction has been introduced by Ritt [8]; second, the non-differential notion of reduced base has its counterpart as *auto-reduced sets*, i.e. sets, where each element is reduced by all others; third, syzygies (i.e. critical pairs) and corresponding S-polynomials are also defined in the differential case; last, the analogue of Groebner bases are *coherent sets*, i.e. sets that reduce all their S-polynomials to 0.

An *auto-reduced coherent set associated to* a set $\mathcal{L}$ of pdp's is an auto-reduced coherent set $\mathcal{M}$ such that $[\mathcal{M}] \subset [\mathcal{L}]$, and $\mathcal{M}$ reduces all pdp's in $\mathcal{L}$ to 0. Computationally, such an associated set is obtained by introducing the critical pairs one after the other, while keeping the set under construction auto-reduced. An algorithm by F. Boulier is given in [1, 2]. Classical noetherianity arguments used in the commutative case to prove termination of algorithms do not extend to the differential case. Instead, an order is defined on auto-reduced coherent sets, and the following theorem ensures the termination of Boulier's algorithm.

THEOREM 1. *There is no infinite decreasing sequence of auto-reduced sets.*

As already mentioned, the author's algorithm is crucially based on the potential cancellation of a certain polynomial H. The following definitions are needed to explain how this polynomial is introduced. They also play an important rôle in the definition of a complete system. Recall that Ritt's reduction relies on an order on the indeterminates $\partial_\alpha y_i$. The *leader* v_P of a pdp P is the highest indeterminate that occurs in it. This notion is the analogue of head terms in usual, non-differential Groebner bases theory. Now, the *initial* I_P of P is the coefficient in $v_P^{\deg P}$ and the *separant* S_P of P is the common initial of all derivatives of P. Finally, given a set $\mathcal{A}$ of pdp's, write $S_\mathcal{A}$ and $H_\mathcal{A}$ for the product of the separants of these pdp's and the product of the initials and separants of these pdp's respectively.

3. Differential extensions by formal power series

The author's crucial assumption is that the f_i's are uniquely defined by systems of PDE's and finite sets of initial conditions at the origin.

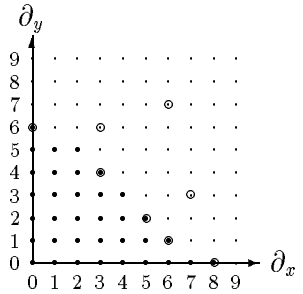


FIGURE 1. To be *under a stairs*

An indeterminate $\partial_\alpha y_i$ is *under the stairs* of a set $\mathcal{A}$ of pdp's if it is the derivative of the leader of no element of this set. An example in the case when $R = \mathbb{C}[x, y]$ and with a single function f is graphically treated on Figure 1: assume the leaders of the elements of a set $\mathcal{A}$ to be $\partial_x^8 f$, $\partial_x^7\partial_y^3 f$, $\partial_x^6\partial_y f$, $\partial_x^6\partial_y^7 f$, $\partial_x^5\partial_y^2 f$, $\partial_x^3\partial_y^4 f$, $\partial_x^3\partial_y^6 f$, and $\partial_y^6 f$ (large framed circles on the figure). The indeterminates under the stairs of $\mathcal{A}$ are then $f, \ldots, \partial_x^8 f$, $\partial_y f, \ldots, \partial_x^6\partial_y f$, $\partial_y^2 f, \ldots, \partial_x^5\partial_y^2 f$, $\partial_y^3 f, \ldots, \partial_x^4\partial_y^3 f$, $\partial_y^4 f, \ldots, \partial_x^3\partial_y^4 f$, $\partial_y^5 f, \ldots, \partial_x^2\partial_y^5 f$, $\partial_y^6 f$ (smaller plain circles on the figure).

When the set of derivatives that are under the stairs of a set $\mathcal{A}$ of pdp's is finite, this set is called a *closed set*. The idea is that a closed set makes it possible, under some assumptions, to recursively compute the values at the origin of all derivatives, provided that the values at the origin of all derivatives under the stairs are given. A *complete system* consists of a closed auto-reduced coherent set $\mathcal{A}$ together with a finite set IC of initial conditions (the values at the origin of the derivatives that are under the stairs), with the additional property that for all $A \in \mathcal{A}$, $A(f)(0) = 0$ but $S_A(f)(0) \neq 0$. These conditions make it possible to compute all values at the origin of all derivatives. This yields the following very old theorem [4, 7].

THEOREM 2. *For any given complete system $(\mathcal{A}, IC)$, there exists a single m-tuple of formal power series which are solutions of $\mathcal{A}$ and which satisfy the initial conditions IC. This tuple is computable, i.e. each coefficient of each f_i is computable.*

More precisely, the coefficients of an f_i are given by a recursive algorithm. Moreover, it is easily proved that each coefficient continuously depends on the initial condition IC, viewed as element of a finite dimensional vector space.

4. Justification for the algorithm

Henceforth, the formal power series f_i are assumed to be defined by a fixed complete system $(\mathcal{A}, IC)$, and the ring $\mathcal{R}$ is assumed to be effective. The problem is to test whether $P_i(f) = 0$ for all P_i in a given set $\{P_1, \ldots, P_t\}$ of pdp's in $\mathcal{R}\setminus R$. This is equivalent to testing whether f is a solution of the system $\{\mathcal{A}, P_1, \ldots, P_r\}$. Boulier's algorithm, which was alluded to before, first reduces the problem to computing with auto-reduced coherent sets, as will be detailed below. Let $\mathcal{B}$ be an auto-reduced coherent set associated with $\{\mathcal{A}, P_1, \ldots, P_t\}$, i.e. a set that satisfies $[\mathcal{B}] \subset [\mathcal{A}, P_1, \ldots, P_t]$, and $Q \xrightarrow{\mathcal{B}} 0$ for all $Q \in \{\mathcal{A}, P_1, \ldots, P_t\}$.

Return into pseudo-reduction: given a set $\mathcal{Q}$ of pdp's, let $H_\mathcal{Q}$ be the product of the initials and separants of the elements of $\mathcal{Q}$ and $S_\mathcal{Q}$ the product of all separants only. Given an ideal $\mathfrak{I}$ which is not necessarily a differential ideal and a pdp H, let $\mathfrak{I} : H^\infty$ denote the set of all pdp's P for which there exists a $\nu \in \mathbb{N}$ such that $H^\nu P \in \mathfrak{I}$. This set is actually an ideal and $P \xrightarrow{\mathcal{Q}} 0$ is equivalent to $P \in \mathcal{Q} : H_\mathcal{Q}^\infty$ [2, 5]. With this notation, it is clear that

$$[\mathcal{B}] \subset [\mathcal{A}, P_1, \ldots, P_t] \subset [\mathcal{B}] : H_\mathcal{B}^\infty . \tag{2}$$

Therefore if $H_{\mathcal{B}}(f) \neq 0$, then when f vanishes at all the elements of $\mathcal{B}$ it vanishes at all the P_i's, so that the problem reduces to testing whether $\mathcal{B}(f) = 0$. Otherwise $H_{\mathcal{B}}(f) = 0$ and the problem reduces to testing whether f is a solution of the system $\{\mathcal{A}, P_1, \ldots, P_r, H_{\mathcal{B}}\}$. Provided that the test for $\mathcal{B}(f) = 0$ is effective, this yields a recursive algorithm that terminates because of Theorem 1. Two cases have to be considered, according to the value of $H_{\mathcal{B}}(f)(0)$.

Regular case. This corresponds to the case when $H_{\mathcal{B}}(f)(0) \neq 0$. For each $B \in \mathcal{B}$, $B(f)$ is a formal power series, which is zero if and only if $\partial_\alpha B(f)(0) = 0$ for all derivation ∂_α (including the identity). A rather technical theorem [6] reduces the problem to considering only finitely many members of this infinite set. Because of the non-nullity of $H_{\mathcal{B}}(f)(0)$, the values at the origin of all the $\partial_\alpha B(f)$'s are polynomials in the $\partial_\beta B(f)(0)$ for β's such that $v_{\partial_\beta B}$ is under the stairs of $\mathcal{A}$, that is for a finite number of initial conditions. More precisely, $\mathcal{B}(f) = 0$ if and only if all these $\partial_\beta B(f)(0)$ equal 0. Since the zero-test in $\mathcal{R}$ is assumed to be effective, this solves the problem in the regular case.

Semi-singular case. This corresponds to the case when $H_{\mathcal{B}}(f)(0) = 0$ while $H_{\mathcal{B}}(f) \neq 0$. Once again, the initial problem on the P_i's reduces to testing $\mathcal{B}(f) = 0$, but the algorithm developed in the regular case cannot be applied as is. An explicit formula for f in terms of the initial conditions IC shows that f depends continuously on IC. The initial conditions IC provide values of the $\partial_\alpha f_i$ for all α such that $\partial_\alpha y_i$ is under the stairs of $\mathcal{A}$. So IC can be viewed as a vector c of a finite dimensional space. Call R' the ring of polynomials $k[x_1, \ldots, x_n, \partial_\alpha y_i]$ where the α's are such that $v_{\partial_\alpha y_i}$ is under the stairs of $\mathcal{A}$ and the $\partial_\alpha y_i$'s are viewed as indeterminates. Let now W be the variety defined by the ideal $\mathfrak{I}$ of R' generated by the $\partial_\alpha B$'s such that $\partial_\alpha B$ is under the stairs of $\mathcal{B}$, and W' the variety defined by $H_{\mathcal{B}} = 0$. The regular case dealt with the implication $c \in W \setminus W' \Longrightarrow \mathcal{B}(f) = 0$. In the current case, the following theorem [6] reduces the problem to computing with algebraic varieties.

THEOREM 3. *Let c be initial conditions such that the system $(\mathcal{A}, IC)$ is complete and $H_{\mathcal{B}}(f) \neq 0$. Then $\mathcal{B}(f) = 0 \Longleftrightarrow c \in \overline{W \setminus W'}$.*

This condition is tested by computing a Groebner bases for the radical of the ideal $\mathfrak{I} : H_{\mathcal{B}}^{\infty}$ using an algorithm described in [2], and testing if each polynomial of the constructed base vanishes at c.

The previous justification yields the algorithm that was outlined before.

Bibliography

[1] Boulier (F.), Lazard (D.), Ollivier (F.), and Petitot (M.). – Representation for the radical of a finitely generated differential ideal. In *Proceedings ISSAC'95*. pp. 158–166. – Association for Computing Machinery, 1995.

[2] Boulier (François). – *Étude et implantation de quelques algorithmes en algèbre différentielle.* – Thèse, Université de Lille, April 1994.

[3] Denef (J.) and Lipshitz (L.). – Power series solutions of algebraic differential equations. *Mathematische Annalen*, vol. 267, 1984, pp. 213–238.

[4] Janet (M.). – Systèmes d'équations aux dérivées partielles. *Journal de Mathématiques*, vol. 8, n° 3, 1920.

[5] Kolchin (E. R.). – *Differential Algebraic Groups.* – Academic Press, New York, 1973.

[6] Péladan-Germa (Ariane). – Testing identities of series defined by algebraic partial differential equations. In *Actes de AAECC'11. Lecture Notes in Computer Science*, pp. 393–407. – Springer-Verlag, 1995. Proceedings AAECC'11, Paris, 1995.

[7] Riquier. – *Les systèmes d'équations aux dérivées partielles.* – Gauthier-Villars, Paris, 1910.

[8] Ritt (Joseph Fels). – *Differential Algebra.* – A.M.S., 1950, *A.M.S. Colloquium*, vol. XXXIII.

Automatic Asymptotics

Joris van der Hoeven
École polytechnique

November 21, 1994

[summary by Bruno Salvy]

Joris van der Hoeven sets up an ambitious research program of automating the derivation of asymptotic expansions in various contexts.

The problem has already been solved in several cases. For exp-log functions—functions obtained from a variable x and the set of rational numbers $\mathbb{Q}$ by closure under field operations and the application of exp and log—John Shackell gave a procedure in [5] which he extended in [6] to Liouvillian functions. In [7], he showed how to handle composition and in [4] B. Salvy and J. Shackell showed how to compute an expansion of $y(x)$ subject to $F(y) = x$ for F an exp-log function. Asymptotic expansions for differential algebraic equations were also made effective by J. Shackell in [8], however the algorithm in this case pays for its generality by an exponential complexity with respect to the order [9]. Another approach to the exp-log function problem was used by D. Gruntz to implement the new `limit` function in Maple [2].

While J. Shackell and several others have based their work on the theory of Hardy fields [3], the approach followed by J. van der Hoeven is inspired by Écalle's theory of transseries [1]. Informally, there are two main ingredients in this work. The first one consists in computing with asymptotic scales suitable for exponentiation and logarithm (so-called *normal bases*, see below). The second one consists in working simultaneously with expansions in these scales and a handle on the exact full information related to them. This handle (named *algorithmic multiseries*) makes it possible to compute more terms of an expansion whenever necessary and to invoke an oracle for zero-equivalence of functions in order to prevent indefinite cancellation. More precise definitions are as follows.

Definition 1. An asymptotic scale is a finite ordered set $\{g_1, \ldots, g_n\}$ of positive unbounded exp-log functions such that $\log g_i = o(\log g_{i+1})$, for $i = 1, \ldots, n-1$.

Definition 2. Let g be a positive unbounded exp-log function. A *multiseries* with respect to g is a formal sum

$$f = \sum_{\alpha \in S} f_\alpha g^\alpha,$$

the coefficients f_α being exp-log functions and the support $S \subset \mathbb{R}$ having finitely generated support:

$$S = \alpha_1 \mathbb{N} + \alpha_2 \mathbb{N} + \cdots + \alpha_k \mathbb{N} + \beta,$$

where the α_i's are strictly positive real numbers and $\beta \in \mathbb{R}$.

Definition 3. A multiseries expansion with respect to $\{g_1, \ldots, g_n\}$, is a multiseries with respect to g_n, where each coefficient can recursively be expressed as a multiseries with respect to $\{g_1, \ldots, g_{n-1}\}$, each multiseries with respect to g_1 having constant coefficients.

DEFINITION 4. A *normal basis* is an asymptotic scale $\{g_1, \ldots, g_n\}$ satisfying the following conditions

(1) $g_1 = \log_k x$ ($k \in \mathbb{N}$), where $\log_k$ denotes the logarithm iterated k times ($\log_0 x = x$);
(2) $\log g_i \in \mathbb{R}[[g_1; \ldots; g_{i-1}]]$, for $2 \leq i \leq n$.

DEFINITION 5. A multiseries with respect to a normal basis $\{g_1, \ldots, g_n\}$ is said to be *algorithmic* when it converges to a function, and for any $k \in \mathbb{N}^\star$ its first k coefficients with respect to g_n can be computed and are themselves algorithmic (constants being algorithmic).

The theorem J. van der Hoeven aims at proving in various contexts of asymptotic expansions is that there always exists a normal basis and (under suitable restrictions) algorithmic multiseries. He has proved this theorem for exp-log functions [10], where the inversion problem and algebraic equations have been at least partially treated too. Here is a theorem from [10].

THEOREM 1. *Schanuel's conjecture implies that the field of real algebraic exp-log functions is an automatic expansion field.*

An automatic expansion field is a field where normal bases and multiseries with respect to them can be computed for any element. Schanuel's conjecture is related to the zero-equivalence problem for constants. Algebraic differential equations and the zero-equivalence problem in a general setting should be treated in [11].

Bibliography

[1] Écalle (Jean). – *Introduction aux fonctions analysables et preuve constructive de la conjecture de Dulac.* – Hermann, Paris, 1992, *Actualités mathématiques.*

[2] Gruntz (Dominik). – *On Computing Limits in a Symbolic Manipulation System.* – PhD thesis, ETH, Zürich, 1995.

[3] Rosenlicht (Maxwell). – Hardy fields. *Journal of Mathematical Analysis and Applications*, vol. 93, 1983, pp. 297–311.

[4] Salvy (Bruno) and Shackell (John). – Asymptotic expansions of functional inverses. In Wang (Paul S.) (editor), *Symbolic and Algebraic Computation.* pp. 130–137. – ACM Press, 1992. Proceedings of ISSAC'92, Berkeley, July 1992.

[5] Shackell (John). – Growth estimates for exp-log functions. *Journal of Symbolic Computation*, vol. 10, December 1990, pp. 611–632.

[6] Shackell (John). – Limits of Liouvillian functions. – Preprint, 1991.

[7] Shackell (John). – Extensions of asymptotic fields via meromorphic functions. – Preprint, 1992.

[8] Shackell (John). – Rosenlicht fields. *Transactions of the American Mathematical Society*, vol. 335, n° 2, 1993, pp. 579–595.

[9] Shackell (John) and Salvy (Bruno). – *Asymptotic Forms and Algebraic Differential Equations.* – Research report n° 2319, Institut National de Recherche en Informatique et en Automatique, August 1994. To appear in the *Journal of Symbolic Computation.*

[10] Van der Hoeven (Joris). – *General algorithms in asymptotics I: Gonnet and Gruntz's algorithm & II: Common operations.* – Technical Report n° LIX/RR/94/10, École polytechnique, Palaiseau, France, July 1994.

[11] Van der Hoeven (Joris). – *Asymptotique Automatique.* – PhD thesis, École polytechnique, Palaiseau, France, 1996. To appear.

Normal Bases and Canonical Rational Form (Over Finite Fields)

Daniel Augot
INRIA – Projet CODES

January 23, 1995

[summary by François Morain]

1. Introduction

Let $k = \mathbb{F}_q$ be the finite field with q elements (q a prime power p^r, r any nonnegative integer). For the basic properties of finite fields, as well as an introduction to normal bases, etc., we urge the unfamiliar reader to read [3, 4, 5].

Let A be a k-linear operator of k^n and call M the associated matrix, that is an element of $\mathcal{M}_n(k)$. The aim of this talk is to introduce the so-called *Shift-Hessenberg form* of M (SHS form for short) and describe its properties. In particular, it will be shown that there exists a fast algorithm for computing H.

Having the SHS form of M enables us to solve several problems. First, we can find cyclic vectors for A and therefore find a normal basis for $\mathbb{F}_{q^n}$ over $\mathbb{F}_q$. We can also find the minimal polynomial of M. Moreover, if we have the factorisation of the characteristic polynomial of M, we can compute the characteristic subspaces of A and get the *Frobenius form* (a.k.a. *rational canonical normal form*) of M. Only the first of these – the computation of cyclic vectors – will be described in this summary. Details can be found in [1].

In the sequel, we will restrict to the case where the characteristic polynomial of A is squarefree, hence equal to the minimal polynomial of A.

2. Cyclic vectors and companion matrices

Let $P(X)$ be a monic polynomial of degree n with coefficients in k:

$$P(X) = X^n + \sum_{i=0}^{n-1} p_i X^i.$$

It is easy to see that $P(X)$ is the characteristic polynomial of the so-called *companion matrix*

$$C_P = \begin{pmatrix} 0 & 0 & \cdots & 0 & -p_0 \\ 1 & 0 & \cdots & 0 & -p_1 \\ 0 & 1 & \cdots & 0 & -p_2 \\ & \ddots & & & \\ 0 & & \cdots & 1 & -p_{n-1} \end{pmatrix}.$$

Let A be a linear operator over k and let $P_A(X)$ denote its minimal polynomial.

Definition 1. If v is a vector in k^n, the minimal polynomial of A relatively to v is the lowest degree nonzero polynomial $P_v(X)$ such that $P_v(A)v = 0$.

DEFINITION 2. A vector v is called *cyclic* if and only if $P_v(X) = P_A(X)$.

One has the following:

THEOREM 1. *Every linear operator A has a cyclic vector.*

In the case where P_A is equal to the characteristic polynomial, and if v is a cyclic vector, the matrix of A in the basis $(v, Av, \ldots, A^{n-1}v)$ is a companion matrix.

Let M be the matrix of A and call C the companion matrix of its characteristic polynomial. The problem we want to solve is how to compute a cyclic vector as fast as possible. We will perform this operation in several steps: the first one is the computation of the Shift-Hessenberg form of M, noted H, and the second is finding C from H.

3. The Shift-Hessenberg form

PROPOSITION 1. *Let M be an $n \times n$ matrix of $\mathcal{M}_n(k)$. There exists a matrix H similar to M of the form:*

$$\begin{pmatrix}
0 & 0 & \times & & & & \times & & & \times \\
1 & 0 & \times & & & & \times & & & \times \\
0 & 0 & \times & & & & \times & & & \times \\
0 & 1 & 0 & & & & \times & & & \times \\
0 & 0 & & 1 & & & \times & & & \times \\
 & & & & \ddots & & \cdots & & & \cdots \\
 & & & & & 1 & \times & & & \times \\
 & & & & & & 0 & & & \times \\
 & & & & & & & 1 & & \times \\
 & & & & & & & & \ddots & \cdots \\
 & & & & & & & & & \times
\end{pmatrix}.$$

The matrix H is called the Shift-Hessenberg form *of M (SHS for short). Computation of H requires $O(n^3)$ elementary operations in k.*

PROOF. Do as in Gauss reduction, but starting from the sub-diagonal. If there is no non-zero element in the first column, below the sub-diagonal, then do nothing. Otherwise, assuming that it is $M_{1,2}$ (permuting lines if needed), eliminate all non-zero entries of this column. At the end of the process, we end up with a matrix of the above form.

The cost of this algorithm is very close to that of Gaussian reduction, that is $O(n^3)$. □

It is clear, that when we can find a pivoting element for each column, we end up with a companion matrix. More generally, any SHS matrix can be written as

$$H = \begin{pmatrix}
H_{B_1,B_1} & H_{B_1,B_2} & \cdots & H_{B_1,B_m} \\
0 & H_{B_2,B_2} & \cdots & H_{B_2,B_m} \\
\vdots & & \ddots & \vdots \\
0 & \cdots & 0 & H_{B_m,B_m}
\end{pmatrix} \tag{1}$$

where m is an integer called the *parameter* of H and where each H_{B_i,B_j} is a companion matrix. It can be shown that the minimal polynomials of the H_{B_i,B_j} are pairwise coprime.

4. From the SHS form to the companion form

The key of the algorithm is the following Lemma [1]

LEMMA 1. *Let A be any block-triangular matrix with two blocks:*

$$A = \begin{pmatrix} A_{B_1,B_1} & A_{B_1,B_2} \\ 0 & A_{B_2,B_2} \end{pmatrix}.$$

For $i = 1, 2$, let $f_i(X)$ be the minimal polynomial of A_{B_i,B_i}. Assume f_1 and f_2 are relatively prime. Let v_{B_i} be a cyclic vector for A_{B_i,B_i}. Let h_2 be such that $h_2(X)f_2(X) \equiv 1 \bmod f_1(X)$. Then a cyclic vector for A is given by

$$v = \begin{pmatrix} u_{B_1} \\ v_{B_2} \end{pmatrix}$$

where

$$u_{B_1} = h_2(A_{B_1,B_1})\left((f_2(A)u_{B_2})_{|B_1} - v_{B_1}\right).$$

The computation of v can be done in $O(n^3)$ field operations.

Now suppose we are given H in the form of (1). If $m = 1$, H is already a companion matrix. If $m = 2$, the preceding Lemma applies. When $m > 2$, there exist two strategies. The first one is to compute a cyclic vector for the last two blocks, replacing these blocks by a companion matrix, and so on, until the whole matrix is companion. The second one is to split H in the form of

$$\begin{pmatrix} H_{B_1,B_1} & H_{B_1,B_2} \\ 0 & H_{B_2,B_2} \end{pmatrix} \tag{2}$$

such that the sizes of H_{B_1,B_1} and H_{B_2,B_2} are kept under control. These can be chosen such that either H_{B_1,B_1} is a single companion block of size $\geq 2n/3$ or both matrices have size $\leq 2n/3$. This leads to two deterministic algorithms. The first one is iterative and has cost $O(n^3 + n^2m^2)$; the second one is recursive and has cost $O(n^3)$. We note that on average, the parameter m is $O(\log n)$.

All these algorithms have been implemented in AXIOM and give very encouraging running times.

5. Normal bases

DEFINITION 3. Let K be a finite extension of degree n of k. An element $\alpha \in K$ is *normal* if and only if

$$K = \mathrm{Vect}_k(\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{n-1}}).$$

If α is normal, then $(\alpha, \alpha^q, \alpha^{q^2}, \ldots, \alpha^{q^{n-1}})$ is called a *normal basis*.

Using a normal basis is particularly useful when computing powers of elements, since this is readily done via a cyclic shift:

$$(a_0, a_1, \ldots, a_{n-1})^q = (a_{n-1}, a_0, \ldots, a_{n-2}).$$

Moreover, it is easy to construct a multiplication table for k by precomputing the quantities $\alpha \times \alpha^{q^i}$ for all i. We can see this as follows. Write

$$c = \left(\sum_{i=0}^{n-1} a_i\alpha^{q^i}\right)\left(\sum_{i=0}^{n-1} b_i\alpha^{q^i}\right) = \left(\sum_{i=0}^{n-1} c_i\alpha^{q^i}\right).$$

Then $c_0 = F_0(a, b)$ is a bilinear symmetric form. Using $c^{q^{n-1}} = a^{q^{n-1}}b^{q^{n-1}}$ we deduce that

$$(a_1\alpha + a_2\alpha^q + \cdots)(b_1\alpha + b_2\alpha^q + \cdots) = c_1\alpha + \cdots$$

or $c_1 = F_0(a^\sigma, b^\sigma)$ where σ denotes the shift operation. We see that computing all c_i's needs only one matrix operation, followed by conjugation.

It is easy to see that:

PROPOSITION 2. *Let $\pi : x \mapsto x^q$ denote the Frobenius automorphism. Then α is normal if and only if α is a cyclic vector for π.*

Using the results of the preceding sections, and noting that the minimal polynomial of π is $X^n - 1$, that is squarefree for $(n, p) = 1$, we get

THEOREM 2. *We can find a normal element in deterministic time $O(n^3 + n^2 \log q)$, where the last term accounts for the computation of a matrix representing π.*

This result improves upon earlier results by von zur Gathen and Giesbrecht who gave a probabilistic algorithm in $O\tilde{(}n^2 \log q)$ (using fast polynomial multiplication) or $O(n^3 \log q)$ without fast multiplication, and a deterministic algorithm running in time $O(n^4 + n^2 \log q)$.

It is possible to treat along the same lines the case where $n = p^k$.

Bibliography

[1] Augot (Daniel) and Camion (Paul). – *On the computation of minimal polynomial, cyclic vectors and the Frobenius form.* – Research Report n° 2006, INRIA, August 1993.

[2] Giesbrecht (Mark William). – *Nearly optimal algorithms for canonical matrix forms.* – PhD thesis, University of Toronto, 1993.

[3] Lidl (Rudolf) and Niederreiter (Harald). – *Finite Fields.* – Addison-Wesley, 1983, *Encyclopedia of Mathematics and its Applications*, vol. 20.

[4] McEliece (Robert). – *Finite fields for computer scientists and engineers.* – Kluwer Academic Publishers, 1988, *Kluwer international series in engineering and computer science.*

[5] Menezes (Alfred J.). – *Applications of Finite Fields.* – Kluwer Academic Publishers, 1993.

[6] Ozello (Patrick). – *Calcul exact des formes de Jordan et de Frobenius d'une matrice.* – Thèse, Université Scientifique Technologique et Médicale de Grenoble, 1987.

[7] von zur Gathen (J.) and Giesbrecht (M.). – Constructing normal bases in finite fields. *Journal of Symbolic Computation*, vol. 10, 1990, pp. 547–570.

Factoring Polynomials Over Finite Fields

Daniel Panario
University of Toronto

January 23, 1995

[summary by Reynald Lercier]

Let $q = p^m$ where p is a prime and $m \in \mathbb{N}^*$, let f be a monic univariate polynomial of degree n in $\mathbb{F}_q[X]$; this talk surveys known algorithms to find the complete factorization $f = f_1^{e_1} \cdots f_r^{e_r}$ where the f_i's are monic distinct irreducible polynomials and where $e_i \in \mathbb{N}^*$ for $i \in \{1, \ldots, r\}$. This problem plays an important role in various fields like computer algebra, cryptography, number theory, coding theory, ...

First, we present the main ideas behind Berlekamp's algorithm (Section 1). Then, we give a general factoring algorithm composed of three stages (squarefree, distinct-degree and equal-degree factorization) which provides a framework for several other algorithms (Section 2). Finally, we outline best known asymptotic complexity, current bottlenecks and recent results (Section 3).

To compare these algorithms, the unit cost will be multiplication in $\mathbb{F}_q$. Moreover, the arithmetic considered is fast for polynomial algorithms and classical for linear algebra.

NOTATION. Throughout the summary, we let $O^{\sim}(n) = O(n \log^k(n))$ where k is a constant.

1. Berlekamp algorithm

Berlekamp's ideas [1] lead to an efficient algorithm to factorize a polynomial f with no repeated factors. Let R be the polynomial ring $\mathbb{F}_q[X]/(f)$ and R_i be the polynomial rings $\mathbb{F}_q[X]/(f_i)$ for $i \in \{1, \ldots, r\}$, then

$$R \simeq R_1 \times \cdots \times R_r$$

by the Chinese Remainder Theorem. We concentrate now on the Frobenius map Φ on R given by

$$\begin{aligned} \Phi : R &\longrightarrow R, \\ h &\longmapsto h^q. \end{aligned}$$

The set of fixed points of Φ is

$$B = \{h \in R, h^q = h\}$$

and again by the Chinese Remainder Theorem, we have

$$B \simeq \mathbb{F}_q^r.$$

This means that B is a $\mathbb{F}_q$ vector space of dimension r, the number of irreducible polynomials f_i. Berlekamp proved the following theorem.

THEOREM 1. *Let $h \in B$, then*

$$f = \prod_{\alpha \in \mathbb{F}_q} \gcd(f, h - \alpha).$$

In fact, we obtain a non trivial factorization of f for $\deg h \geq 1$ (if $\deg h = 0$, we obtain f multiplied by 1). Definition 1 outlines an important property of B.

DEFINITION 1. A set $S = \{h_1, \ldots, h_s\}$ is called a separating set for f if for any two distinct irreducible factors f_i and f_j, there exists $h_k \in S$ and $\alpha_k \in \mathbb{F}_q$ such that $h_k - \alpha_k$ is divisible by f_i, but not f_j.

Theorem 2 connects Theorem 1 with Definition 1.

THEOREM 2. *Let $\{1, v_2, \ldots, v_r\}$ be a polynomial basis of B, then $\{v_2, \ldots, v_r\}$ is a separating set for f.*

These results finally lead to the following algorithm whose complexity is $O^{\sim}(n^3 + qn^2)$.

(1) Form the matrix of the mapping $\Phi - \mathrm{Id}$ with respect to the basis $\{1, X, \ldots, X^{n-1}\}$ of R;
(2) Obtain a basis $\{1, v_2, \ldots, v_r\}$ of $B = \ker(\Phi - \mathrm{Id})$ using Gaussian elimination;
(3) Factor f computing

$$f = \prod_{\alpha \in \mathbb{F}_q} \gcd(f, v_2 - \alpha)$$

and refine the partial factorization successively using $v_3, \ldots, v_r$ until the complete factorization of f is obtained.

2. A general factoring algorithm

We focus now on a different approach for factoring polynomials over finite fields. It is a general method that breaks the problem into three subproblems.

Square-free factorization: Find monic square-free pairwise relatively prime polynomials $g_1, \ldots, g_n$ such that

$$f = g_1 g_2^2 \cdots g_n^n.$$

Distinct degree factorization: Split a square-free polynomial into polynomials the irreducible factors of which have the same degree.
Equal degree factorization: Completely factor a polynomial the irreducible polynomial factors of which have the same degree.

2.1. Square-free factorization. This subproblem is solved easily by more or less computing the gcd of f with its derivative f'. More rigourously, a possible algorithm is the following [3].

$i := 1$; $R := 1$; $a := f$; $b := f'$; $c := \gcd(a, b)$; $w := a/c$;
while $c \neq 1$ do
    $y := \gcd(w, c)$; $z := w/y$; $R := R * z^i$; $i := i + 1$; $w := y$; $c := c/y$
$R := R * w^i$;
return(R)

This method has cost $O^{\sim}(n)$.

2.2. Distinct degree factorization. This subproblem is solved by Theorem 3 [4].

THEOREM 3. *For every $i \in \mathbb{N}$, the product of all monic irreducible polynomials over $\mathbb{F}_q$ whose degrees divide i is equal to $X^{q^i} - X$.*

This theorem leads to the following algorithm which returns a n-tuple $(g_1, \ldots, g_n)$ where each polynomial g_i contains all the factors of degree i of f.

(1) Set $h_0 = X$ and $f_0 = f$;
(2) For $i = 1, \ldots, n$, do
  - Compute $h_i = h_{i-1}^q \bmod f$;
  - Compute $g_i = \gcd(h_i - X, f_{i-1})$ and $f_i = f_{i-1}/g_i$;
(3) Return $(g_1, \ldots, g_n)$.

The main cost of this algorithm is the computation of X^{q^i} for $i = 1, \ldots, n$. By repeated squaring, this cost is $O^{\sim}(n^2 \log q)$. A better way of computing this quantity consists in using an algorithm to iterate the Frobenius map. At first we compute $X^q \bmod f$ and then go doubling evaluating $X^{q^i} \bmod f$ in X^q to obtain $X^{q^{i+1}}$. So $X^{q^2} \bmod f = X^q(X^q) \bmod f$, $X^{q^3} \bmod f = X^{q^2}(X^q) \bmod f$, and so on. The cost of this method is $O^{\sim}(n^2 + n \log q)$

2.3. Equal degree factorization. The probabilistic algorithm we are going to describe to find the r irreducible factors $f_1, \ldots, f_r$ of degree d of a polynomial f of degree $n = dr$ in $\mathbb{F}_q$ with q odd is due to Cantor-Zassenhaus [2].

If there exists a polynomial $c \in \mathbb{F}_q[X]$ such that $c \bmod f_i = 0$ and $c \bmod f_j \neq 0$ for $1 \leq i < j \leq r$, then $\gcd(c, f)$ splits f. To take advantage of this idea, we choose at random a polynomial a of $\mathbb{F}_q[X]/(f)$. Then the polynomials $a_i = a \bmod f_i$ for $1 \leq i \leq r$ are independent and uniformly distributed elements in $\mathbb{F}_q[X]/(f_i)$. So

$$a_i^{\frac{q^d-1}{2}} \equiv \pm 1 \bmod f_i$$

with probability $1/2$ each. Consequently, $a^{\frac{q^d-1}{2}} - 1$ does not factor f with probability $2(1/2)^r$. This idea leads to the following algorithm which returns one factor of f or FAIL.

(1) Choose $a \in \mathbb{F}_q[X]/(f)$ at random;
(2) Compute $g = \gcd(a, f)$. If $g \neq 1$, then return g;
(3) Compute $b = a^{\frac{q^d-1}{2}} - 1 \bmod f$;
(4) Compute $g = \gcd(b, f)$. If $g \neq 1$, then return g else return FAIL.

Its cost is $O^{\sim}(n^2 \log q)$.

3. Recent results

New results improve the scheme described in Section 2.

- J. von zur Gathen and V. Shoup [7]: Distinct degree factorization: $O^{\sim}(n^2 + n \log q)$; equal degree factorization: $O^{\sim}(n^{1.7} + n \log q)$.
- E. Kaltofen and V. Shoup announced in [6]: Distinct degree factorization: $O(n^{1.815} \log q)$ asymptotically but $O(n^{2.5} + n \log q)$ in practice; equal degree factorization: $O(n^2 \log n + n \log q)$ in practice.

Nevertheless, the main cost remains the computation of X^{q^i} for $i = 1, \ldots, n$. Furthermore, other recent results must be cited.

- Evdokimov (1993): A deterministic algorithm, quasi polynomial time $(n^{\log n} \log q)^{O(1)}$.
- Niederreiter [5]: A new deterministic algorithm.

– Kaltofen-Lobo (1994): Randomized Berlekamp with Wiedemann's linear solver, in time $O^{\sim}(n^2 + n \log q)$.

Bibliography

[1] Berlekamp (E. R.). – Factoring polynomials over large finite fields. *Mathematics of Computation*, vol. 24, n° 111, 1970, pp. 713–735.

[2] Cantor (D. G.) and Zassenhaus (H.). – A new algorithm for factoring polynomials over finite fields. *Mathematics of Computation*, vol. 36, 1981, pp. 587–592.

[3] Geddes (Keith O.), Czapor (Stephen R.), and Labahn (George). – *Algorithms for Computer Algebra*. – Kluwer Academic Publishers, 1992.

[4] Lidl (Rudolf) and Niederreiter (Harald). – *Finite Fields*. – Addison-Wesley, 1983, *Encyclopedia of Mathematics and its Applications*, vol. 20.

[5] Niederreiter (H.). – Factoring polynomials over finite fields using differential equations and normal bases. *Mathematics of Computation*, vol. 62, 1994, pp. 819–830.

[6] Shoup (V.). – A new polynomial factorization algorithm and its implementation. – 1994. Preprint.

[7] von zur Gathen (J.) and Shoup (V.). – Computing Frobenius map and factoring polynomials. *Computational Complexity*, vol. 2, 1992, pp. 187–224.

Computation of the Integral Basis of an Algebraic Function Field and Application to the Parametrization of Algebraic Curves

Mark van Hoeij
University of Nijmegen

June 7, 1995

[summary by Laurent Bertrand]

Abstract

A new algorithm [1] for computing an integral basis of an algebraic function field is presented. This algorithm is then applied to the computation of parametrizations of algebraic curves of genus zero [2].

1. Computation of the integral basis

Let L be an algebraically closed field of characteristic zero and x be transcendental over L. Let y be algebraic over $L(x)$ with minimal polynomial f of degree n with respect to y. We suppose that y is integral over $L[x]$, so f is monic over $L[x]$. Let C be the algebraic curve defined by the equation

$$f(X,Y) = 0$$

and let $L(C)$ be the function field

$$L(C) = L(x,y) = L(X)[Y]/(f(X,Y)).$$

A function of $L(C)$ is called *integral* if it satisfies a monic irreducible polynomial with coefficients in $L[x]$. The integral closure Θ of $L[x]$ in $L(C)$ is the set of all integral functions. It is also the set of all functions with no finite pole, and it is a free module of rank n over $L[x]$. An *integral basis* is then a set $\{b_0, \ldots, b_{n-1}\}$ of elements of $L(C)$ such that

$$\Theta = L[x]b_0 + \cdots + L[x]b_{n-1}.$$

The algorithm presented here computes an integral basis with all its elements in $K(x,y)$ where K is a given subfield of L containing all the coefficients of f.

1.1. Algorithm. The algorithm can be described as follows. We look for an integral basis of the form $\{b_0, \ldots, b_{n-1}\}$ such that b_i is a polynomial of degree i in y with coefficients in $K(x)$. Moreover b_0 can be chosen equal to 1. The integral basis is computed step by step. Suppose that

$$\{b_0, \ldots, b_{d-1}\}$$

have been computed, then we compute b_d such that

$$L[x]b_0 + \cdots + L[x]b_d = \{a \in \Theta : \deg(a) \leq d\}$$

and $\deg(b_d) = d$ as follows:

(1) let b_d be yb_{d-1};

(2) let $V = \{a \in \Theta : \deg(a) \leq d\} \setminus L[x]b_0 + \cdots + L[x]b_d$;
while $V \neq \emptyset$ do
(a) choose $a \in V$ such that $a = (a_0b_0 + \cdots + a_db_d)/k$ with $a_0, \ldots, a_d$ and k in $K[x]$ and $a_d = 1$;
(b) substitute b_d by a.

In order to compute an element a satisfying the conditions of (a), the author applies the result saying that $x - \alpha$ appears in the denominator k if and only if C has a singularity on the line $x = \alpha$. After that, for computing the a_i's, Puiseux expansions are used and also bounds for these expansions and for the degree of the denominator. The issue is the resolution of a linear system.

2. Application to the parametrization of algebraic curves

Here f is supposed to be irreducible of degree n with respect to y. The curve C is the projective algebraic curve defined by f. Let F be the homogenization of f. It means that $F(X, Y, Z)$ is the polynomial of smallest degree such that $f = F(X, Y, 1)$. A parameter p is a function generating $L(C)$, i.e., every function in $L(C)$ can be written as a rational funtion in p. It is in fact a function with only one pole which is of order 1 on C. A parametrization of C is a pair $(X(t), Y(t))$ of rational functions such that $f(X(t), Y(t)) = 0$ and $L(X(t), Y(t)) = L(t)$.

Curves allowing parametrizations are called rational curves. They are in fact curves of genus 0. The aim of this algorithm is to compute when it is possible a parametrization of a given curve, using the algorithm for computing an integral basis presented before.

2.1. Algorithm. The algorithm for computing a parametrization is the following:

(1) Compute a parameter p;
(2) Express x and y as rational functions in p.

For the computation of a parameter, divide the projective plane in two disjoint parts A and B. Compute a function P with only one pole of multiplicity 1 in $A \cap C$. Then compute a function Q with no pole in $A \cap C$ and such that $P + Q$ has no pole in $B \cap C$. (For that, the computation of an integral basis is used). Then a parameter is $P + Q$.

The last thing to do is to express x and y as rational functions in p by computing appropriated resultants.

The computation of integral basis can also be used to compute the genus of a curve or the Weierstrass normal form of a curve of genus 1, see [1, 3].

Bibliography

[1] van Hoeij (Mark). – An algorithm for computing an integral basis in an algebraic function field. *Journal of Symbolic Computation*, vol. 18, 1994, pp. 353–363.

[2] van Hoeij (Mark). – Computing parametrizations of rational algebraic curves. In *Symbolic and Algebraic Computation*. ACM, pp. 187–190. – New York, 1994. Proceedings ISSAC'94, Oxford, England.

[3] van Hoeij (Mark). – An algorithm for computing the Weierstrass normal form. In *Symbolic and Algebraic Computation*. ACM. – ACM Press, 1995. Proceedings ISSAC'95, Montreal, Canada.

Symbolic Computation of Hyperelliptic Integrals and Arithmetic in the Jacobian

Laurent Bertrand
Université de Limoges

June 7, 1995

[summary by Gaétan Haché]

1. Introduction

The interest of this talk is the integration of hyperelliptic functions. The technique used for such integration follows the usual pattern for the integration of algebraic function developed by R. H. Risch, B. M. Trager, J. H. Davenport and a representation of divisors over hyperelliptic curves due to D. G. Cantor. For example, we want to compute the integral

$$\int_2^3 \frac{3x^5 - x + 2}{(x^2 - x^5 - x + 1)\sqrt{x^5 + x - 1}}\, dx. \tag{1}$$

If the exact value of this integral is needed, then one normally computes a primitive. In this case it is equal to

$$\log\left(\frac{x + \sqrt{x^5 + x - 1}}{x - \sqrt{x^5 + x - 1}}\right).$$

If we set

$$y^2 = x^5 + x - 1, \tag{2}$$

then one can consider the integral

$$\int \frac{3x^5 - x + 2}{(x^2 - x^5 - x + 1)y}\, dx$$

over the algebraic function field of the affine curve

$$\mathcal{C} = \{(a, b) \in \mathbb{A}^2(\mathbb{C}) : b^2 - a^5 - a + 1 = 0\}.$$

2. Integration over function field of curves

The general setup is the following. Let K be a field of characteristic 0, $\overline{K}$ an algebraic closure of K and $F \in K[x, y]$ an absolutely irreducible polynomial (that is F is irreducible over $\overline{K}$). We consider the function field

$$K(\mathcal{C}) = K(x)[y]/\langle F \rangle$$

where $\mathcal{C}$ is the curve defined by $F(x, y) = 0$.

Definition 1. A function H is said to be an *elementary primitive* of $h \in K(\mathcal{C})$ if $H' = h$ and if H can be written from functions of $K(\mathcal{C})$ using combinations of logarithms, exponentials and algebraic expressions.

In the previous example

$$h = \frac{3x^5 - x + 2}{(x^2 - x^5 - x + 1)y},$$

the curve has for equation

$$y^2 - x^5 - x + 1 = 0$$

and h has for elementary primitive

$$\log\left(\frac{x+y}{x-y}\right).$$

We want to answer the following questions:

(1) Does a function $h \in K(\mathcal{C})$ have an elementary primitive $H = \int h\,dx$ over K ?
(2) If so, what is this primitive?

Risch have shown that if H is an elementary primitive over K then

$$H = v_0 + \sum_{i=1}^{k} c_i \log(v_i) \tag{3}$$

where $v_0 \in K(\mathcal{C})$, $c_i \in \overline{K}$ and $v_i \in \overline{K}(\mathcal{C})$. The algebraic part v_0 is computed using Hermite's algorithm and the logarithmic part is done using Risch's algorithm.

Following is a short history of integration of algebraic functions.

1833: Liouville's principle; gives the form of the elementary primitive;
1872: Hermite's algorithm; allows the computation of the algebraic part;
1970: Risch's algorithm; allows the computation of the logarithmic part. Needs arithmetic over divisors of function fields and principality test;
1981–1984: Davenport and Trager algorithms; first implementable algorithms.

3. Special case: hyperelliptic curves

In his thesis, B. Trager gives an algorithm which solves the previous questions in the general cases. The work of L. Bertrand studies the case where $\mathcal{C}$ has genus $g \geq 2$ (hyperelliptic) and $K(\mathcal{C})$ is a quadratic extension of $K(x)$ with $x \in K(\mathcal{C})$ transcendental over K. Let L be the function field of genus g of the curve $\mathcal{C}$ defined by the equation

$$y^2 = f(x) \tag{4}$$

where $f(x)$ is square free of degree m. In this case, the computation of the logarithmic part of the primitive is reduced to the computation of the primitive of the following type

$$\int \omega \qquad \text{where} \qquad \omega = \frac{P(x)}{Q(x)y}\,dx$$

with $P, Q \in K[x]$ such that $\gcd(Q', Q) = \gcd(P, Q) = \gcd(f, Q) = 1$. To the differential ω is associated some zero degree divisors $D_1, D_2, \ldots, D_k$ over the normalized of the affine curve $\mathcal{C}$ defined by (4). A necessary condition for the primitive to be elementary is that all these divisors are torsion divisors, that is there exist m_i, $(i = 1, \ldots, k)$, such that $m_i D_i$ are principal. Then the functions $v_i \in K(\mathcal{C})$ such that $(v_i) = n_i D_i$ are candidates to verify (3).

4. Representation of divisors

For quadratic extensions, L. Bertrand has developed an algorithm which is much more efficient than Trager's one. This is because the test of principality is greatly improved by the use of a simpler representation of divisors over such quadratic extensions. Following is an overview of such representations. Two cases are considered:

- $m = 2g + 1$ and $\mathcal{C}$ has a unique point at infinity;
- $m = 2g + 2$ and $\mathcal{C}$ has exactly two points at infinity.

Case $m = 2g + 1$. Any divisor D of degree 0 may be written

$$D = \sum_{i=1}^{k} n_i P_i - \sum_{i=1}^{k} n_i P_\infty. \tag{5}$$

It is represented by two polynomials

$$[a(x), b(x)]$$

where

- $a(x) = \prod_{i=1}^{k} (x - x_i)^{n_i}$;
- for all i, $\nu_P(y - b(x)) \geq n_i$;
- $\deg b < \deg a$.

Case $m = 2g + 2$. Any divisor D of degree 0 may be written

$$D = \sum_{i=1}^{k} n_i P_i + n_{\infty+} P_{\infty+} + n_{\infty-} P_{\infty-}$$

with $\sum_{i=1}^{k} n_i + n_{\infty+} + n_{\infty-} = 0$ and the representation of D is given by

$$[a(x), b(x), \delta]$$

where a and b are defined in the same manner as in the case $m = 2g + 1$ and where $\delta = n_{\infty+} - n_{\infty-}$.

5. Arithmetic in the Jacobian

To test if a divisor D is a torsion divisor, one computes the order of D in the Jacobian over several well chosen finite prime fields (note that over finite fields, any zero degree divisor is a torsion divisor since the Jacobian of the curve is finite). Using the outcome of these order computation, one can decide if the divisor D is a torsion divisor or not.

The computation of the order of a divisor is done by performing a principality test of lD for $l = 1, 2, \ldots$ until we find l such that lD is principal. To do so in an efficient way, fast arithmetic computation over the Jacobian is needed. Following is an overview of how it is done using the representation of divisors by two polynomials (for more details see [1]). Let D be a divisor represented by $[a, b]$. Then $-D = [a, -b] - (a)$. Let D_1 and D_2 be two divisors represented by $[a_1, b_1]$ and $[a_2, b_2]$ respectively. Then

$$D_1 + D_2 = \left[\frac{a_1 a_2}{d^2}, \frac{h_1 a_1 b_2 + h_2 a_2 b_1 + h_3(b_1 b_2 + f)}{d} \mod a \right] + (d) \tag{6}$$

where

$$d = \gcd(a_1, a_2, b_1 + b_2) = h_1 a_1 + h_2 a_2 + h_3(b_1 + b_2).$$

A notion of reduced divisor is considered and the principality test relies on a theorem stating that a reduced divisor D is principal if and only if $D = [1, 0]$. Using the arithmetic over the Jacobian, one can compute for any divisor D an equivalent reduced divisor D_0, that is such that $D = D_0 + (h)$ for some function $h \in K(\mathcal{C})$. In the case where $m = 2g + 2$, a similar notion of reduced divisor is used, and a reduced divisor D is principal if and only if $D = [1, 0, 0]$.

Bibliography

[1] Cantor (D. G.). – Computing in the Jacobian of an hyperelliptic curve. *Mathematics of Computation*, vol. 48, n° 177, 1987.

Part 3

Asymptotic Analysis

Asymptotics of Mahler Recurrences: Binary Partitions Weighted by the Number of Summands

Philippe Dumas
INRIA, Rocquencourt

December 5, 1994

[summary by Hsien-Kuei Hwang]

Abstract

The asymptotic behaviour of the number of binary partitions (summands being powers of two) weighted by the number of summands is investigated. The methods of proof relies on the Mellin-Perron formula.

1. Introduction

Consider the generating function

$$f(z) = \sum_{n\geq 0} a_n z^n = \prod_{k\geq 0} \frac{1}{1-\rho z^{2^k}}, \qquad (\rho > 0), \tag{1}$$

which satisfies the functional equation of Mahler type:

$$f(z)(1-\rho z) = f(z^2). \tag{2}$$

The general problem of interest here is the asymptotic behaviour of the sequence a_n, as $n \to +\infty$. Different values of ρ give rise to different behaviours of a_n. The simplest case is when $\rho > 1$. One easily deduces from (2)

$$a_n = f\left(\rho^{-2}\right)\rho^n + O\left(\rho^{n/2}\right), \qquad (n \to +\infty).$$

When $\rho = 1$, a_n represents the number of partitions of n into summands which are powers of two. Asymptotics of a_n were originally studied by C. L. Siegel and by K. Mahler, and later by N. G. de Bruijn [2]. The principal methods used by De Bruijn are Mellin transform (without explicit mention) and the saddle-point method. His result is

$$\log a_{2n} = \log a_{2n+1} = \frac{1}{2\log 2}\left(\log\frac{n}{\log n}\right)^2 + \left(\frac{1}{2} + \frac{1}{\log 2} + \frac{\log\log 2}{\log 2}\right)\log n - \\ - \left(1 + \frac{\log\log 2}{\log 2}\right)\log\log n + \varpi\left(\frac{\log n - \log\log n}{\log 2}\right) + O\left(\frac{(\log\log n)^2}{\log n}\right),$$

as $n \to +\infty$, where $\varpi(u)$ is a periodic function of period 1 whose Fourier expansion is explicited. Another approach (with weaker error term) to similar problems by Pennington [7] proceeds along Ingham's Tauberian theorem.

The problem becomes very complex when $\rho = e^{it}$, $0 < |t| \leq \pi$, t real. A study of the various behaviours of a_n using elementary methods is contained in van der Hoeven's DEA memoir.

This talk is concerned with the case when $0 < \rho < 1$. As individual term are highly irregular, one considers the summatory function of a_n.

THEOREM 1. *Let a_n be defined in (1) with $0 < \rho < 1$ and set $F_n = \sum_{0 \le k \le n} a_k$. Then F_n satisfies*

$$F_n = P(\log_2 n) n^\alpha + O\left(n^{\alpha+\varepsilon-1/2}\right), \qquad (\varepsilon > 0), \tag{3}$$

as $n \to +\infty$, where $\alpha = -\log(1-\rho)/(\log 2) > 0$ and $P(u)$ is a periodic function of u whose mean value is approximately given by

$$\frac{e^{-\lambda/\log 2}}{\Gamma(\alpha+1)}(1-\rho)^{\gamma/(\log 2)-1/2}, \qquad \textit{where} \qquad \lambda = \sum_{k\ge 1} \frac{\log k}{k}\rho^k.$$

2. Proof

To derive (3), one starts from the Mellin-Perron formula:

$$\sum_{1\le j\le n} a_j = \frac{1}{2i\pi}\int_{c-i\infty}^{c+i\infty} \frac{n^{s+1}}{s}\varphi(s)\mathrm{d}s, \tag{4}$$

where c is taken to be larger than the abscissa of absolute convergence of the Dirichlet series

$$\varphi(s) := \sum_{j\ge 1} a_j j^{-s}.$$

To apply (4), one requires the analytic continuation of φ to a larger half-plane (than its original domain of analyticity) and the magnitude of growth of the continued function at $\sigma \pm i\infty$.

The abscissa of convergence of φ is determined by the growth order of F_n, cf. [8, §9.14]. From the defining equation (1), one readily obtains the recurrence

$$\begin{cases} a_0 &= 1; \\ a_n &= \rho a_{n-1} + a_{n/2}, \end{cases} \tag{5}$$

with the convention that $a_x = 0$ when $x \notin \mathbb{Z}$. From this, one deduces the following relations for F_n,

$$\begin{cases} F_0 &= 1; \\ F_n &= \rho F_{n-1} + F_{n/2} + F_{(n-1)/2}, \end{cases}$$

again with the convention that $F_x = 0$ when $x \notin \mathbb{Z}$. From this last recurrence, one can verify, by induction, that

$$F(n) = O\left(n^\alpha\right).$$

Thus, by [8, §9.15], the abscissa of absolute convergence of the Dirichlet series $\varphi(s)$ is not greater than α. A probabilistic argument concerning the distribution of the number of summands in a binary partition permits to show that the abscissa of absolute convergence of $\varphi(s)$ is in fact less than α.

The analytic continuation of φ can be computed by (5) and the technique used in [1]. One thus obtains

$$(1-\rho-2^{-s})\varphi(s) = \rho + \rho g(s),$$

where

$$g(s) = \sum_{j\ge 1} a_j\left(j^{-s} - (j+1)^{-s}\right) = \sum_{k\ge 1}\binom{s+k-1}{k}(-1)^{k+1}\varphi(k+s).$$

The second expression provides the required analytic continuation of φ to the whole s-plane.

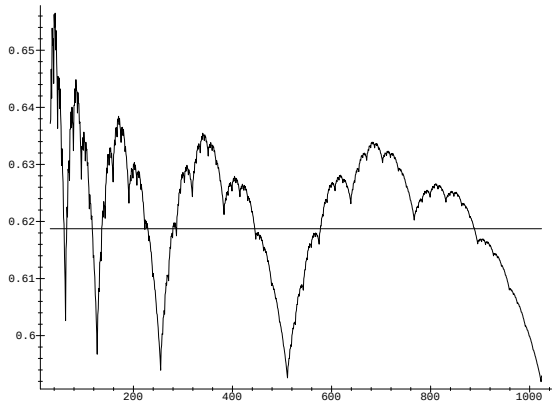


FIGURE 1. The reduced sequence f_n/n^α associated to $\rho = 1/2$ contrasted with the mean value of the periodic function $P(v)$.

The remaining analysis follows closely the lines for the number of odd numbers in Pascal triangle in [4]. The expression for the approximate mean-value of $P(u)$ is obtained by Mellin transform starting from

$$\log f(e^{-t}) = \sum_{j \geq 0} \log \frac{1}{1 - \rho e^{-2^j t}} \sim -\alpha \log t + p(\log_2 t) + q(t),$$

as $t \to 0^+$, where p is a 1-periodic function and q is an entire function. The key point is the strong correspondence (see [5]) between the asymptotic behaviour of a function in the vicinity of 0 and the singularities of its Mellin transform. This provides a precise estimation for the residues of the Mellin transform of $f(e^{-t})$, which is $\Gamma(s)\varphi(s)$. The theorem of residues is then applied, hence the theorem stated. See Figure 1 for an illustration.

3. Concluding remarks

The number of unrestricted partitions (whose summands are positive integers) weighted by the number of summands has generating function

$$\sum_{n \geq 0} p(n) z^n = \prod_{k \geq 1} \frac{1}{1 - \rho z^k}.$$

The corresponding asymptotics have been thoroughly studied in the literature. The case $\rho = 1$ leads to the famous Hardy-Ramanujan-Rademacher formula, and other cases were completed by Wright [9].

The methods presented in this talk, adapted from [4], become more or less standard and are powerful enough to be applicable to other problems, like q-multiplicative functions [6], the Goldberg problem of determining the asymptotic behaviour of the coefficients

$$[z^n] \exp\left(\sum_{j \geq 0} z^{2^j} \right),$$

divide-and conquer recurrences, etc.

Bibliography

[1] Allouche (Jean-Paul) and Cohen (Henri). – Dirichlet series and curious infinite products. *Bulletin of the London Mathematical Society*, vol. 17, 1985, pp. 531–538.

[2] de Bruijn (N. G.). – On Mahler's partition problem. *Indagationes Mathematicae*, vol. 10, 1948, pp. 210–220. – reprinted from *Koninklijke Nederlandse Academie van Wetenschappen, Proceedings*.

[3] Dumas (Ph.). – Asymptotics of Mahlerian recurrences: binary partitions weighted by the number of summands, 1996. To appear.

[4] Flajolet (P.), Grabner (P.), Kirschenhofer (P.), Prodinger (H.), and Tichy (R. F.). – Mellin transforms and asymptotics: digital sums. *Theoretical Computer Science*, vol. 123, n° 2, 1994, pp. 291–314.

[5] Flajolet (Ph.), Gourdon (X.), and Dumas (Ph.). – Mellin transforms and asymptotics : Harmonic sums. *Theoretical Computer Science*, vol. 144, n° 1–2, June 1995, pp. 3–58.

[6] Grabner (P.). – Completely q-multiplicative functions: the Mellin transform approach. *Acta Arithmetica*, vol. 65, 1993, pp. 85–96.

[7] Pennington (W. B.). – On Mahler's partition problem. *Annals of Mathematics*, vol. 57, 1953, pp. 531–546.

[8] Titchmarsh (E. C.). – *The theory of functions*. – Oxford University Press, Oxford, 1939, second edition.

[9] Wright (E. M.). – Asymptotic partition formulae: (II) weighted partitions. *Proceedings of the London Mathematical Society*, vol. 36, 1933, pp. 117–141.

Oscillating Rivers

Franck Michel
Université de Nice

February 6, 1995

[summary by Jacques Carette]

Abstract

An oscillating river is an oscillating asymptotic solution of an ordinary differential equation where there is, at infinity, an exponential concentration of solutions of the differential equation. The aim of this talk is to present a few cases where such can be proved to occur.

1. Introduction

F. and M. Diener have recently studied some cases of solutions of ordinary differential equations which are exponentially close to each other at infinity [1, 2, 3]. This new type of attractor was baptised "fleuve", or "river" a name suggested by the corresponding phase portraits (see Figure 1). If one considers a scalar equation

$$\frac{dY}{dX} = \sum_{j=0}^{n} P_j(X) Y^j$$

where the $P_j(X)$ are finite sums of rational powers of X with complex coefficients, then there exist effective methods to ascertain the presence of rivers. The associated solutions are either attractive, in which case there is an infinity of solutions which share the same asymptotic behaviour, else they are repulsive, in which case there is a unique asymptotically unstable solution. These rivers generally possess divergent asymptotic expansions in fractional powers of X, but they are always Gevrey.

2. The Periodic Model

The results established for the $P_j(X)$ in the class defined above, are still valid if these functions possess a pole at plus infinity, but not if one of them has an essential singularity. As the figures show, this is not an obstacle for such rivers to occur. However, F. Michel believes that it is the periodic structure of the functions considered which makes the phenomenon possible. This leads to the study of the following model:

$$\frac{dY}{dX} = \sum_{i \in I} a_i(X) X^{m_i} Y^{n_i} \tag{1}$$

where $m_i \in \mathbb{Q}, n_i \in \mathbb{N}, I$ is a finite set, $a_i(X) \in \mathcal{C}$, where $\mathcal{C}$ is the algebra of $\mathcal{C}^\infty$ periodic functions of fixed period τ.

The asymptotic behaviour of these rivers does not follow from those of the previous case (that we shall refer to as the polynomial case). In fact, there will be three cases, depending on a parameter c which depends on the m_i and the n_i. The method used to prove the theorems will be to give

qualitative results about solutions that bound (above and below) the trajectories of interest. This is to be contrasted to the methods used for the polynomial case, which were singular perturbations and non-standard techniques.

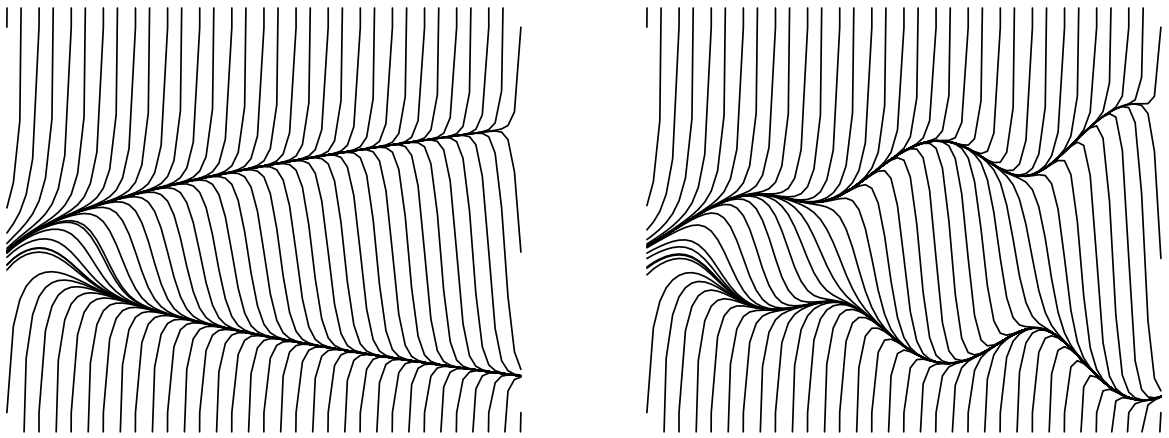


FIGURE 1. Rivers: $Y' = Y^2 - X$ and $Y' = Y^2 - (1 + \sin X/2)X$.

3. Notation

Even though the results are straightforward to state in vague terms, the exact formulation needs a number of preliminary concepts to be defined.

DEFINITION 1. If $k(X) \in \mathcal{C}$ is a non-zero function and $Y(X)$ is a real function defined on a neighbourhood of $+\infty$ and $r \in \mathbb{Q}$ then we say that $Y(X)$ is asymptotic to $k(X)X^r$ (which we denote $Y(X) \sim k(X)X^r$) if

$$Y(X) = k(X)X^r + o(X^r), \qquad X \to \infty.$$

Note that if a function is asymptotic to $k(X)X^r$ then the term $k(X)$ is necessarily unique which makes $\mathcal{C}$ an appropriate space for studying the asymptotic behaviour of solutions of our model.

As is usual in dealing with differential equations, crucial information is contained in the Newton polygon $\mathcal{P}$, the convex hull of the set of horizontal half-lines going left from the points (m_i, n_i). A rational number r will be called a co-slope of $\mathcal{P}$ if r is the slope of one of the normals to the segments of $\mathcal{P}$.

Note $Q(S, X, Y) = \sum_{i \in I} a_i(S) X^{m_i} Y^{n_i}$ where $a_i(X) \in \mathcal{C}$ and are not identically zero. The number $\bar{a}_i$ is the average of a_i over one period, and $\tilde{a}_i(X) = a_i(X) - \bar{a}_i$. We also generalise this notation to any function of X, by which we mean that a bar indicates an average and a tilde the zero-average translation.

Let $r \in \mathbb{Q}$, then we define $\mu_0 = \max(m_i + rn_i)$ and $c = 1 - r - \mu_0$. For the slope r, c measures the attraction (or repulsion) of the associated solution, if it exists. Let q be the smallest positive integer such that $r - n/q$ for $n \in \mathbb{N}$ takes on all values of $m_i + rn_i$ for $i \in I$ and the value $r - 1$. From these values we can define $p = cq$ and $\mu_n = \mu_0 - n/q$. Furthermore, we set

$$Q_r^n(S, X, Y) = \sum_{m_i + rn_i = \mu_n} a_i(S) x^{m_i} Y^{n_i}, \qquad \bar{Q}_r^n(X, Y) = \sum_{m_i + rn_i = \mu_n} \bar{a}_i x^{m_i} Y^{n_i}.$$

We will only discuss the various Q_r^0 functions here, but the other functions can be used to determine the successive terms in the asymptotic expansion. We denote by a $'$ the derivative of the above functions with respect to Y. The differential equation

$$\frac{dY}{dX} = Q_r^0(X, 1, Y) \tag{2}$$

will be important. Finally, note by $(*)$ an expansion of the form

$$\sum_{i \geq 0} \alpha_i(X) X^{r - i/q}$$

for $\alpha_i(X) \in \mathcal{C}$. This is the model used for an asymptotic expansion.

4. Results

Each of the following three theorems has two sub-cases, sub-case (a) corresponds to the attractive case, and sub-case (b) to the repulsive case.

THEOREM 1. *Let $r \in \mathbb{Q}$, $k(X) \in \mathcal{C}$, $k \neq 0$ be such that*

(1) *r is a co-slope of $\mathcal{P}$;*
(2) *$c > 1$;*
(3) *$\forall X$, $Q_r^0(X, 1, k(X)) = 0$;*
(4) *(a) $\forall X$, $(Q_r^0)'(X, 1, k(X)) < 0$, or (b) $\forall X$, $(Q_r^0)'(X, 1, k(X)) > 0$.*

Then there exists a series of the type $(*)$ which is a formal solution of (1) with $\alpha_0(X) = k(X)$. Furthermore, there exists an infinite number of solutions asymptotic to $k(X)X^r$ in the attractive case, and a unique one in the repulsive case. The series $(*)$ is the asymptotic expansion of those solutions.

Conditions (3) and (4) express that $k(X)X^r$ is the first term in the asymptotic expansion for a trajectory with constant 0 derivative along that trajectory. The first condition is to insure that the function Q_r^0 is not reduced to one term (where only the function $k(X) \equiv 0$ would satisfy the third condition). The cases in Condition (4) indicate whether the branch of the solution we are considering is attractive or repulsive. The fact that $c > 1$ indicates that geometrically the nearby solutions oscillate along with the trajectory considered.

THEOREM 2. *Let $r \in \mathbb{Q}$, $k \in \mathbb{R}$, $k \neq 0$ be such that*

(1) *r is a co-slope of $\mathcal{P}$;*
(2) *$0 < c < 1$;*
(3) *$\bar{Q}_r^0(1, k) = 0$;*
(4) *(a) $(\bar{Q}_r^0)'(1, k) < 0$, or (b) $(\bar{Q}_r^0)'(1, k) > 0$.*

Then there exists a series of the type $(*)$ which is a formal solution of (1) with $\alpha_0(X) = k(X)$. Furthermore, there exists an infinite number of solutions asymptotic to kX^r in the attractive case, and a unique one in the repulsive case. The series $(*)$ is the asymptotic expansion of those solutions.

Since $0 < c < 1$, the rivers do not oscillate starting at the first approximation, and thus it is necessary to look at the averaged function $\bar{Q}_r^0$ instead, but the meaning of the third and fourth conditions are essentially the same as in the previous theorem.

THEOREM 3. *Let $r \in \mathbb{Q}$, $k(X) \in \mathcal{C}$, $k(X) \neq 0$ be such that*

(1) *r is a co-slope of $\mathcal{P}$;*
(2) *$c = 1$;*

(3) *$k(X)$ is a periodic solution of (2);*
(4) *(a)* $\overline{(Q_r^0)'(X,1,k(X))} < 0$, *or (b)* $\overline{(Q_r^0)'(X,1,k(X))} > 0$.

Then there exists a series of the type $(*)$ which is a formal solution of (1) with $\alpha_0(X) = k(X)$. Furthermore, there exists an infinite number of solutions asymptotic to $k(X)X^r$ in the attractive case, and a unique one in the repulsive case. The series $(*)$ is the asymptotic expansion of those solutions.

The case $c = 1$ is an intermediate situation: there are oscillations of the type $k(X)X^r$, but $k(X)$ does not correspond exactly to the oscillations of the trajectory. When it exists, we observe that it is generally out of phase and of smaller amplitude.

We will then call *oscillating river* the solutions described by each of the preceding theorems.

5. Example

Let us consider the equation $Y' = (Y^2 - (2 + \sin(X)))X^\alpha$. The first condition in all theorems implies that necessarily $r = 0$. From this, we can calculate $c = 1 + \alpha$. Thus, if $\alpha > 0$, the first theorem gives that we have a river asymptotic to $\pm(2 + \sin(X))^{1/2}$, if $1 < \alpha < 0$, the second one gives rivers asymptotic to $\pm\sqrt{2}$ and if $\alpha = 0$, the last theorem leads us to search for periodic solutions of a periodic equation, where one can consult the large literature on this subject.

6. Proof Ideas

To prove that there exists solutions asymptotic to $k(X)X^r$, the notion of *tunnels* is used. If there exists $X_0, \nu_-, \nu_+ \in \mathbb{R}$ with $\nu_- < \nu_+$ such the right hand side of (1) is positive for all $X \geq X_0$ for $Y = \nu_-$ and negative for $Y = \nu_+$, then the set $\{(X,Y) \mid X \geq X_0, \nu_- < Y < \nu_+\}$ is called a tunnel. In the appropriate coordinates, the hypotheses imply easily that such tunnels exist, which then force the existence of the asymptotic solutions. Technical computations show the existence of a formal series solution. And finally, a few more arguments with tunnels and leading term comparisons allow us to conclude that this series is actually an asymptotic series expansions for the solutions shown previously to exist.

Bibliography

[1] Diener (F.). – Singularités des équations différentielles, Dijon 1985. *Astérisque*, vol. 150-151, 1987, pp. 59–66.

[2] Diener (F.). – Propriétés asymptotiques des fleuves. *Comptes-Rendus de l'Académie des Sciences*, vol. 302, 986, pp. 55–58.

[3] Diener (F.) and Diener (M.). – Fleuves 1-2-3 : mode d'emploi. In Diener (M.) and Wallet (G.) (editors), *Mathématiques finitaires et analyse non standard.* pp. 209–216. – Publications Mathématiques de l'Université de Paris VII, 1989.

[4] Michel (Franck). – Fleuves oscillants. *Bulletin of the Belgian Mathematical Society*, vol. 2, 1995, pp. 127–141.

Analytical Approach to Some Problems Involving Order Statistics

Wojciech Szpankowski
Purdue University

June 16, 1995

[summary by Danièle Gardy]

Abstract

Order statistics, such as the distribution of the maximum of n random variables, are usually studied from a probabilistic point of view. This talk presents an analytical approach that can be applied to a sequence of independent random variables, and to dependent variables. Applications include statistics on digital structures, the analysis of a leader election algorithm, and an extension of probabilistic counting.

1. Order statistics

Let $X_1, X_2, \ldots, X_n$ be a sequence of discrete random variables; the order statistics is the sequence arranged in nondecreasing order: $X_{(1)} \leq X_{(2)} \leq \cdots \leq X_{(n)}$. The classical theory of order statistics takes place in a probabilistic frame; see for example [2] or [12].

Assume that the variables X_i are *exchangeable:* the $n!$ permutations $X_{i_1}, \ldots, X_{i_n}$ have the same joint distribution [3, p. 228].[1] Define $M_n = \max\{X_1, \ldots, X_n\}$ as the maximum of the n variables. By the inclusion-exclusion principle, we have that

$$\Pr\{M_n > k\} = \sum_{r=1}^{n} (-1)^{r+1} \binom{n}{r} \Pr\{X_1 > k, \ldots, X_n > r\}. \tag{1}$$

Define

$$\widehat{F_r}(z) = \sum_{k \geq 0} \Pr\{X_1 > k, \ldots, X_r > k\} z^k; \qquad \widehat{M_n}(z) = \sum_{k \geq 0} \Pr\{M_n > k\} z^k.$$

Then Equation (1) translates on the generating series as

$$\widehat{M_n}(z) = \sum_{r=1}^{n} (-1)^{r+1} \binom{n}{r} \widehat{F_r}(z).$$

Hence the generating function of M_n (or of the rth-ranked variable of the sequence) is expressed by an alternating sum, which suggests that a Mellin-Rice approach to the asymptotics might be successful (see for example [6] for a general introduction to this subject).

[1] Another definition of exchangeable variables might be: for any subsequence $\{i_j\}$ s.t. $1 \leq i_1 < \cdots < i_r \leq n$, $\Pr\{X_{i_1} < x_1, \ldots, X_{i_r} < x_r\} = \Pr\{X_1 < x_1, \ldots, X_r < x_r\}$.

2. The independent case: the probabilistic approach

2.1. Continuous random variables. In the continuous case, the X_i are i.i.d. and continuous; there exist two sequences of normalization constants $\{a_n\}$ and $\{b_n\}$ and a function H s.t. $\lim_{n\to+\infty} \Pr\{(M_n - a_n)/b_n < x\} = H(x)$; then H is the *limiting distribution* of the (normalized) maximum M_n.

The theory of asymptotic distribution of extremes was initiated by Fischer and Tippett in 1928, and further developed by Gnedenko around 1943; see for example the books by Galambos [7] or Resnick [12] or the presentation given by Sweeting in [13]. The behaviour of the tail determines the limiting distribution $H(x)$ (see for example [2, p. 210] or [7, p. 51-52] for complete conditions). The limiting distribution of the normalized maximum sample has one of the following types: *(i)* If $\Pr(X_i > tx)/\Pr(X_i > t) \to x^{-\alpha}$ $(\alpha > 0)$ for $t \to +\infty$, then $H(x) = \exp(-x^{-\alpha})$ for $x > 0$ and $H(x) = 0$ for $x \le 0$; *(ii)* If there exists a finite λ s.t. $\Pr(X_i \le \lambda) = 1$, and $\Pr(X_i > \lambda - \epsilon x)/\Pr(X_i > \lambda - \epsilon) \to x^{\alpha}$ for $\epsilon \to 0$, then $H(x) = \exp(-(-x)^{\alpha})$ for $x < 0$ and $H(x) = 1$ for $x \ge 0$; *(iii)* If $\Pr(X_i > t + xE(X_i - t|X_i > t))/\Pr(X_i > t) \to e^{-x}$ for $t \to +\infty$, then $H(x) = \exp(-e^{-x})$ for all x.

The normalization constants a_n and b_n might be seen respectively as a shift and a scaling factor. They are not necessarily unique: see [2, p. 209] or [12, p. 86] for a discussion of this point; Galambos devotes a whole section of his book to discussing possible choices [7, p. 57–63]. In good cases, a_n and b_n correspond to the limiting mean and variance; see [12, p. 84-85] for conditions that ensure that we can use the mean and variance as scaling factors.

When it is possible to prove conditions on the tail distribution, such as an exponential tail, the asymptotic mean can be computed as: $a_n = \inf\{x : \Pr(X_i \ge x) \le 1/n\}$.

2.2. Discrete random variables. In this case, Anderson [1] (see also [7, p. 120, ex. 8]) gave a necessary condition for the existence of a_n and b_n s.t. the normalized random variable $(M_n - a_n)/b_n$ converges to a non-degenerate limiting distribution: $\Pr(X_i = k)/\Pr(X_i > k) \to 0$, $k \to \infty$.

The existence of a *limiting* distribution is a strong property, which is not always verified; in some cases we can prove a weaker result on the existence of an *asymptotic* distribution, which might not imply a limiting distribution because of fluctuations. An example of this happens when the X_i follow a geometric distribution: $\Pr(X_i = k) = p^k(1-p)$ for $k \ge 0$. Then it is possible to prove that $\Pr\{M_n < \lfloor \log_{1/p} n + m \rfloor\} \sim \exp\left(-p^{m-\{\log_{1/p} n + m\}}\right)$. Because of the fluctuating nature of the fractional part $\{t\} = t - \lfloor t \rfloor$, this expression oscillates between e^{-p^m} and $e^{-p^{m-1}}$.

3. The independent case: the analytical approach

For i.i.d. variables X_i, we have $\Pr\{X_1 > k, \ldots, X_r > k\} = (P\{X_1 > k\})^r$. Hence $\hat{F}^r = X^{\circledR}$, with $X^{\circledR}(z) = \sum_k (\Pr\{X > k\})^r z^k$ standing for r Hadamard products.

When the distribution of the X_i is a sum of geometric distributions, their g.f. is $\hat{X}(z) = a/(1-\rho z)^d$ for some constants a, ρ and d. Now, for integer d, the coefficient $[z^n]\hat{X}(z)$ is equal to $a\binom{n+d-1}{d-1}\rho^n$ and we can compute a uniform approximation of $\hat{F}^r(z)$:

$$\hat{F}^r(z) = \frac{(dr-r)!a^r}{((d-1)!)^r(1-\rho^r z)^{dr-r+1}} + \Theta\left(\frac{1}{(1-\rho^r z)^{(d-1)r}}\right).$$

From now on a Mellin-Rice approach can be used, to obtain

THEOREM 1. *Let $a_n = \log n + (d-1)\log\log n - \log\Gamma(d)$; then for any integer k*

$$\Pr(M_n \le a_n + k) = \frac{1}{\log 1/\rho} e^{-\rho^{k+1+\{a_n\}}}(1 + o(1)).$$

4. Digital structures

4.1. Depth in a trie. Consider a trie built on n independent random uniform binary sequences $S_i, 1 \leq i \leq n$, of 0 and 1. Define D_n as the average depth of an external node [14]; this quantity is related to the number of nodes visited during a successful search. The analysis of D_n leads to considering the length $C_{i,j}$ of the longest prefix common to the sequences S_i and S_j: D_n has the same distribution as $\max\{C_{1,2}, \ldots, C_{1,n}\}$. Although the $C_{i,j}$ are dependent, we still have

$$\Pr\{D_n \geq k\} = \frac{1}{n} \sum_{r=2}^{n} (-1)^r \binom{n}{r} r \Pr\{C_{1,2} \geq k, \ldots, C_{1,r} \geq k\}. \tag{2}$$

The Bernoulli model. In this model, $\Pr\{C_{1,2} \geq k, \ldots, C_{1,r} \geq k\} = (p^r + q^r)^k$. Define $\widehat{G_n}(z) = \sum_{k \geq 0} \Pr\{D_n \geq k\} z^k$; the relation (2) on D_n translates into an equation on $\widehat{G_n}(z)$, which gives after some computations

$$\widehat{G_n}(z) = \frac{1}{2i\pi} \int_{-3/2-i\infty}^{-3/2+i\infty} \frac{n^{-s-1}\Gamma(s+1)}{1 - z(p^{-s} + q^{-s})} ds \left(1 + O\left(\frac{1}{n}\right)\right).$$

In the asymmetric case, D_n follows a normal limiting distribution, with asymptotic mean $\mu_n \sim (1/h)\log n$ and variance $\sigma_n^2 \sim c \log n$; the constant c is $(h_2 - h^2)/h^3$, with $h = -p\log p - q \log q$ and $h_2 = p \log^2 p + q \log^2 q$. The proof relies on Goncharev's condition, characterizing a normal distribution from its g.f.: $\lim_{n \to \infty} e^{-\theta \mu_n/\sigma_n} G_n(e^{\theta/\sigma_n}) = e^{\theta^2/2}$. In the symmetric case ($p = q = 1/2$), the variance is $O(1)$ ($c = 0$), which suggests that Goncharev's condition does not hold and that we cannot expect a normal limiting distribution. Indeed, the asymptotic distribution fluctuates according to the fractional part of $\log_2 n$: $\Pr\{D_n \leq \log_2 n + k\} \sim \exp(-2^{k+1+\{\log_2 n\}})/\log 2$.

The Markovian case. In the Markovian model, the next symbol depends on the previous one only; the probability $p_{i,j}$ of obtaining the letter i after the letter j is given by a matrix P. It is possible to write an equation on the g.f. of the depth D_n and a similar analysis [8] shows that D_n again tends to a normal limiting distribution, with a variance of order $\log n$, except for the symmetric independent model, where the variance is $O(1)$.

4.2. An open problem: height of a trie. The approach outlined in Section 4.1 fails when one considers the height of a trie, defined as the maximal depth of all leaves: $H_n = \max\{C_{i,j}, 1 \leq i < j \leq n\}$. The catch here is that the variables are not exchangeable.

4.3. Depth of a digital search tree. Consider a digital search tree built on n independent keys in the Bernoulli model; as for a trie, let D_n be the average depth of an external node, and define $\mathrm{E}\, B_n(k)$ as the average number of internal nodes at level k. Then $\Pr\{D_n = k\} = \mathrm{E}\, B_n(k)/n$. The generating function $B_n(u) := \sum_{k \geq 0} \mathrm{E}\, B_n(k) u^k$ satisfies the recurrence equation

$$B_{n+1}(u) = 1 + u \sum_{j=0}^{n} \binom{n}{j} p^j q^{n-j} (B_j(u) + B_{m-j}(u)),$$

whose solution can be expressed in terms of $Q_k(u) = \prod_{j=2}^{k+1} (1 - (p^j + q^j)u)$. Again the asymptotic distribution in the symmetric case fluctuates with n, and a central limit theorem can be proved in the asymmetric case [10].

The Lempel-Ziv algorithm for data compression can be modelled by a digital search tree built on independent keys, when the number n of parsed words is known, and its performance can be expressed in terms of parameters of the tree such as the average depth of an external node. A

different model considers that the pertinent information is the length of the sequence to be parsed; again this can be modelled by a digital search tree, now with *dependent* keys. It is possible to prove (see [10]) that the distribution of the length of a random phrase is asymptotically the same as the limiting distribution of the depth in the first model, with a digital search tree built on $m = \lfloor nh \log n \rfloor$ nodes (here again h denotes the entropy of the alphabet: $h = -p \log p - q \log q$).

4.4. A leader election algorithm. The algorithm for the election of a loser, analyzed by Prodinger in [11], can be dealt with in a similar manner [4]. The principle of the algorithm is as follows: At the beginning, all players are active; at each step, the active players throw a coin randomly and independently and the set of active players for the next step is exactly those who throw *tail*, or all the former players if all of them throw *heads*; the algorithm ends when a single player throws *tail*. The number of steps required by the algorithm to choose a loser is the height H_n of the leftmost leaf of a trie. The analysis begins with the study of the Poisson model, where the number of keys follows a Poisson distribution, then goes on to extract the statistics for the Bernoulli model by a *Depoissonization Lemma*.

4.5. Probabilistic counting. This generalization of an algorithm by Flajolet and Martin [5], using an array of integers instead of a bitmap, is presented in [9].

Bibliography

[1] Anderson (C. W.). – Extreme value theory for a class of discrete distributions with applications to some stochastic processes. *Journal of Applied Probability*, vol. 7, 1970, pp. 99–113.

[2] Arnold (B.), Balakrishnan (N.), and Nagaraja (H. N.). – *A first course in order statistics.* – Wiley series in Probability and Statistics, 1992.

[3] Feller (W.). – *An Introduction to Probability Theory and its Applications.* – Wiley & Sons, New York, 1971, vol. 2.

[4] Fill (J.), Mahmoud (H.), and Szpankowski (W.). – *On the distribution for the duration of a randomized leader election algorithm.* – Technical Report n° CSD-TR-95-038, Purdue University, 1995.

[5] Flajolet (P.) and Martin (G. N.). – Probabilistic counting algorithms for data base applications. *Journal of Computer and System Sciences*, vol. 31, n° 2, October 1985, pp. 182–209.

[6] Flajolet (Philippe) and Sedgewick (Robert). – Mellin transforms and asymptotics: finite differences and Rice's integrals. *Theoretical Computer Science*, vol. 144, n° 1–2, June 1995, pp. 101–124.

[7] Galambos (J.). – *The asymptotic theory of extreme statistics.* – Wiley, New York, 1978.

[8] Jacquet (P.) and Szpankowski (W.). – Analysis of digital tries with Markovian dependency. *IEEE Transactions on Information Theory*, vol. 37, n° 5, 1991, pp. 1470–1475.

[9] Kirschenhofer (P.), Prodinger (H.), and Szpankowski (W.). – How to count quickly and accurately: A unified analysis of probabilistic counting and other related problems. In *Automata, Languages, and Programming*, *Lecture Notes in Computer Science*, pp. 211–222. – 1992. Proceedings of the 19th ICALP Conference.

[10] Louchard (G.) and Szpankowski (W.). – Average profile and limiting distribution for a phrase size in the Lempel-Ziv parsing algorithm. *IEEE Transactions on Information Theory*, vol. 41, 1995, pp. 478–488.

[11] Prodinger (H.). – How to select a loser. *Discrete Mathematics*, vol. 120, 1993, pp. 149–159.

[12] Resnick (S. I.). – *Extreme values, regular variation and point processes.* – Springer Verlag, 1987.

[13] Sweeting (T. J.). – On domains of uniform local attraction in extreme value theory. *Annals of Probability*, vol. 13, n° 1, 1985, pp. 196–205.

[14] Szpankowski (W.). – On the height of digital trees and related problems. *Algorithmica*, vol. 6, n° 2, 1991, pp. 256–277.

The Solution to a Conjecture of Hardy

John Shackell
University of Kent, Canterbury

October 10, 1994

[summary by Joris van der Hoeven]

Abstract

John Shackell proves a conjecture of Hardy, which states that the inverse function of $\log\log x \log\log\log x$ is not asymptotic to any exp-log function. In order to prove this, he uses his technique of nested forms.

1. Introduction

Hardy was the first to study systematically the notion of exp-log functions in the context of asymptotic expansions [3, 4]. These functions are built up from $\mathbb{Q}$ or $\mathbb{R}$ by the use of field operations, exponentiation and logarithm. Examples are

$$\exp(x + \log(x^2 + e^x)), \quad \text{and} \quad \log(\log(x^3 + \exp(1995x)) + 2).$$

He established that the sign of any exp-log function is constant in a neighbourhood of infinity. This property makes exp-log functions extremely useful for doing asymptotics. Although many functions one encounters in practice are asymptotic to some exp-log function, Hardy conjectured that this is not the case for the inverse function $\Phi(x)$ of $\log_2 x \log_3 x$, where the index denotes iteration. In other words, Φ is defined by

$$\log\log\Phi(x)\log\log\log\Phi(x) = x.$$

Now it is known since Liouville [5] that the inverse function $\Psi(x)$ of $x\log x$ is not an exp-log function. In his talk Shackell shows how to deduce Hardy's conjecture from this result.

In order to do this, Shackell uses his technique of nested expansions, which was originally designed to construct algorithms for doing asymptotics. Although Shackell also spoke about these issues in his talk, we will only recall the material which is necessary in order to prove Hardy's conjecture. For more details about the algorithmic aspects of asymptotics, we refer to [1, 2, 6, 7, 8, 9, 10, 11, 12].

2. On nested expansions

We start with some definitions. Let f_1 and f_2 be exp-log functions which tend to zero. We say that f_1 and f_2 are *comparable* or of the same asymptotic scale, if there exist positive integers m and n with $f_1 \leq f_2^m$ and $f_2 \leq f_1^n$ (recall that the germs of exp-log functions at infinity form a totally ordered field). The comparability relation is an equivalence relation and we denote the equivalence class of f by $\gamma(f)$. The equivalence classes can be ordered by $\gamma(f_1) > \gamma(f_2)$, if $f_1 \leq f_2^n$ for all positive integers n.

Let us also introduce the concept of z-functions. Such a function is one of the following:

$$\begin{aligned}\mathrm{zexp}_n(t) &= t^{-n}(\exp(t) - 1 - t - \cdots - t^n/n!),\\ \mathrm{zlog}_n(t) &= t^{-n}(\log(1+t) - t - \cdots - (-1)^{n-1}t^n/n),\\ \mathrm{zinv}_n(t) &= t^{-n}(1/(1+t) - 1 + t - \cdots - (-1)^{n-1}t^n),\end{aligned}$$

for any integer $n \geq 0$. If $t_1, \ldots, t_m$ are exp-log functions which tend to zero, we denote by $Z(t_1, \ldots, t_m)$ the set of functions which can be obtained from $t_1, \ldots, t_m$ by using addition, subtraction, multiplication and application of z-functions. Shackell proved the following theorem [8].

THEOREM 1. *Let f be an exp-log function which tends to infinity. Then there exist exp-log functions $t_1, \ldots, t_m$ with $\gamma(t_1) > \cdots > \gamma(t_m)$, such that f can be expressed as $f = \exp_r((k+z)L)$, where $\exp_r$ is the r-th iterated exponential, k is a non zero constant, L is a product of real powers of iterated logarithms, and z belongs to $Z(t_1, \ldots, t_m)$.*

The expression $f = \exp_r((k+z)L)$ is called a nested form of f. More generally, one can recursively compute nested forms for $t_1, \ldots, t_m$. Doing this, one obtains so called nested expansions. Shackell and Salvy have shown how to obtain automatically nested expansions of the functional inverses of exp-log functions [7], modulo suitable hypotheses on exp-log constants.

3. The solution to Hardy's conjecture

Denote by Ψ the inverse function of $x \log x$, and recall that $\Phi = \exp_2 \Psi$ denotes the inverse function of $\log_2 x \log_3 x$. The following lemma is crucial for the proof of Hardy's conjecture.

LEMMA 1. *There is no exp-log function f such that*

$$|f - \Psi| \leq e^{-\delta\sqrt{x}},$$

for all $\delta > 0$.

PROOF. Assume that such a function f exists. It can be shown (using the same notations as in the above theorem), that one can find $z \in Z(t_1, \ldots, t_m)$ such that

$$f = \frac{x}{\log x}(1+z).$$

Now replace all terms in the Laurent series expansion of z in $t_1, \ldots, t_m$, which have equivalence class superior or equal to $\gamma(e^{\sqrt{x}})$ by zero. Let $\hat{z}$ be the series so obtained and denote $\hat{f} = (x/\log x)(1+\hat{z})$. Then it can be shown that $\hat{f}$ is an exp-log function, so that modulo changing δ, we may assume without loss of generality that $\gamma(t_1) < \cdots < \gamma(t_m) < \gamma(e^{\sqrt{x}})$.

Now it is easily seen that

$$|f \log f - x| = |f \log f - \Psi \log \Psi| \leq e^{-\delta'\sqrt{x}},$$

for some suitable δ'. But $f \log f$ and x are both analytic functions in $x, \log x, \log_2 x, t_1, \ldots, t_m$, so that we must have $f \log f = x$. But this is impossible by Liouville's theorem. Hence, we obtained the desired contradiction. □

THEOREM 2. *There does not exist any exp-log function which is asymptotic to the inverse function of $\log_2 x \log_3 x$.*

PROOF. Since $\Psi = x/\log\Psi = x/(\log x - \log_2 \Psi)$, we have

$$\Phi = \exp_2(x/(\log x - \log_4 \Phi)).$$

Now let g be asymptotic to Φ, so that $\log g - \log \Phi = o(1)$. Then

$$\log g/\log \Phi = 1 + o(\log^{-1} \Phi) = 1 + o(\exp((\varepsilon - 1)x/\log x)),$$

for any $\varepsilon > 0$. Hence

$$|\log_2 g - \log_2 \Phi| < \exp^{-\delta\sqrt{x}},$$

for all $\delta > 0$. By the lemma, it follows that $\log_2 g$ cannot be an exp-log function. Hence neither is g. □

This theorem shows that the scale of all exp-log functions is not sufficient to do asymptotic expansions of functional inverses. This shows that one essentially needs more general asymptotic scales, or an alternative way to represent asymptotic series. One of the candidates for such an alternative way of representing series is Shackell's technique of nested expansions.

Bibliography

[1] Geddes (Keith O.) and Gonnet (Gaston H.). – A new algorithm for computing symbolic limits using hierarchical series. In Gianni (P.) (editor), *Symbolic and Algebraic Computation. Lecture Notes in Computer Science*, vol. 358, pp. 490–495. – New York, 1989. Proceedings ISSAC'88, Rome.

[2] Gonnet (Gaston H.) and Gruntz (Dominik). – *Limit Computation in Computer Algebra.* – Technical Report n° 187, ETH, Zürich, November 1992.

[3] Hardy (G. H.). – *Orders of Infinity.* – Cambridge University Press, 1910, *Cambridge Tracts in Mathematics*, vol. 12.

[4] Hardy (G. H.). – Some results concerning the behaviour at infinity of a real and continuous solution of an algebraic differential equation of the first order. *Proceedings of the London Mathematical Society*, vol. 10, 1911, pp. 451–468.

[5] Liouville (J.). – Suite du Mémoire sur la classification des transcendantes et sur les racines de certaines équations en fonction finie explicite des coefficients. *Journal de Mathématiques Pures et Appliquées*, vol. 3, 1838, pp. 523–546.

[6] Salvy (Bruno). – *Asymptotique automatique et fonctions génératrices.* – Ph. D. Thesis, École polytechnique, 1991.

[7] Salvy (Bruno) and Shackell (John). – Asymptotic expansions of functional inverses. In Wang (Paul S.) (editor), *Symbolic and Algebraic Computation.* pp. 130–137. – ACM Press, 1992. Proceedings of ISSAC'92, Berkeley, July 1992.

[8] Shackell (John). – Growth estimates for exp-log functions. *Journal of Symbolic Computation*, vol. 10, December 1990, pp. 611–632.

[9] Shackell (John). – Limits of Liouvillian functions. – Preprint, 1991.

[10] Shackell (John). – Inverses of Hardy L-functions. *Bulletin of the London Mathematical Society*, vol. 25, 1993, pp. 150–156.

[11] Shackell (John). – Rosenlicht fields. *Transactions of the American Mathematical Society*, vol. 335, n° 2, 1993, pp. 579–595.

[12] van der Hoeven (J.). – *Asymptotique automatique.* – PhD thesis, École Polytechnique, France, 1995. In preparation.

Part 4

Analysis of Algorithms and Data Structures

Using Functional Analysis in Average Case Analysis: the Example of the Gauss Reduction Algorithm

Brigitte Vallée
Université de Caen

May 15, 1995

[summary by Pierre Nicodème]

Abstract

The Gaussian algorithm may be viewed as a formal generalization of the Euclidean algorithm: it uses an extension of the real shift operator U used for continued fractions. We study the random variable "number of iterations" L, when the input data are distributed along an initial density, and we describe the evolution of the data while processing the algorithm. The results use spectral properties of a family of Ruelle-Mayer operators $\mathcal{H}_s$ "inverting" the shift operator U. The operator family $\mathcal{H}_s$ defines a unifying framework allowing a common analysis of both Euclid and Gauss algorithms. This work is a generalization of a common work with Hervé Daudé and Philippe Flajolet [2].

1. The Euclidean and Gaussian algorithms

Starting from a lattice in dimension 2, $\mathcal{L} = \mathbb{Z}u \bigoplus \mathbb{Z}v$, with $u, v \in \mathbb{C}$ not collinear, the Gaussian algorithm finds a minimal basis (m, n) in the sense that the triangle built on (m, n) has no obtuse angle. The problem is invariant by similitude $u \mapsto \lambda u$, with $\lambda \in \mathbb{C}$, and therefore the problem on (u, v) is equivalent to the problem on $(1, v/u)$. The triangle built on $(1, z)$ has no obtuse angles iff $z \in \mathcal{B} - \mathcal{D}$, where $\mathcal{B} = \{z, 0 \leq \Re z \leq 1\}$, and $\mathcal{D}$ is the disk of diameter $[0, 1]$.

FIGURE 1. A lattice and two of its bases represented by the parallelogram they span. The first basis is skew, the second one is minimal (reduced).

The Gaussian algorithm is the composition of a succession of transforms of two types: *(i)* inversion S with $S(z) = 1/z$, *(ii)* translation T^{-m} with $T(z) = z + 1$. With $U(z) = 1/z - \lfloor \Re(1/z) \rfloor$, the Gaussian algorithm terminates whenever $U^k(z) \in \mathcal{B} - \mathcal{D}$. Applying a suitable transform T^{-m} with acute basis, so that $\Re(z) \geq 0$, it is readily seen that it suffices to consider cases where $z \in \mathcal{D}$.

The Gaussian algorithm for lattice reduction is a generalization of the Euclidean algorithm for finding the gcd of two integers in the following way:

	Euclid Continued Fractions	Gauss Lattice Reduction
Algorithm	Input: $x \in [0,1[$	input $z \in \mathcal{D}$ = disc of diameter $[0,1[$
	while $x \neq 0$ do $x = 1/x - \lfloor 1/x \rfloor$	while $z \in \mathcal{D}$ do $z = 1/z - \lfloor \Re(1/z) \rfloor$
Termination	terminates on $\mathbb{Q}$ when the input is in $\mathbb{R}$	terminates on $\mathbb{C} \setminus \{\mathbb{R} \setminus \mathbb{Q}\}$ when the input is in $\mathbb{C}$

We are investigating a generalization of a problem set by Gauss around 1800 for the Euclidean algorithm: starting with a density f on $[0,1]$, what is the density $F_k[f]$ after k iterations of U, with $U(x) = 1/x - \lfloor 1/x \rfloor$. The possible antecedents of x are of the form $1/(m+x)$, with $m \geq 1$, and F_k and F_{k+1} are connected by

$$F_{k+1}[f](x) = \sum_{m \geq 1} \frac{1}{(m+x)^2} F_k[f]\left(\frac{1}{m+x}\right). \tag{1}$$

Introducing the operator $\mathcal{G}$, defined by

$$\mathcal{G}_s[f](x) = \sum_{m \geq 1} \frac{1}{(m+x)^s} f\left(\frac{1}{m+x}\right), \tag{2}$$

many properties of the Euclidean algorithm can be expressed in terms of spectral quantities related to the operator $\mathcal{G}_s$ (with s close to 2): the existence of a limit density $F_\infty[f](x) = \frac{1}{\log 2}\frac{1}{1+x}$ corresponds to the dominant eigenvector of $\mathcal{G}_2$ (with eigenvalue $\lambda = 1$); the expectation of the number K_n of iterations of Euclid on p/q verifying $1 \leq p < q \leq N$ is given by $\mathrm{E}[K_N] = \frac{12 \log 2}{\pi^2} \log N + \mathcal{O}(1)$, in tight relationship with $\lambda'(2)$ (with $\lambda(s)$ dominant eigenvalue of $\mathcal{G}_s$).

We will derive from the properties of the $\mathcal{G}$ operators the "stationary" distribution $F_\infty[f]$, and the distribution of the number L of iterations of the Gaussian algorithm along any initial distribution f.

Like the continued fraction expansion of a number under the Euclidean algorithm, with $z_j \in \mathcal{D}$, which implies $\Re(1/z_j) > 1$, we have $z_{j+1} = 1/z_j - m_j$, with $m_j \geq 1$, which is equivalent to $z_j = 1/(m_j + z_{j+1})$, and gives the expansion

$$z_0 = \cfrac{1}{m_1 + \cfrac{1}{m_2 + \cfrac{1}{\ddots \cfrac{}{m_k + z_k}}}}. \tag{3}$$

This expansion terminates as soon as $z_k \in \mathcal{B} - \mathcal{D}$. Then $L(z_0) = k$, $z_0 = h_m(z_k)$ and $h_m(z)$ may be expressed in terms of the continuants $Q_k(m_1, m_2, \ldots, m_k)$ and $P_k(m_1, m_2, \ldots, m_k) = Q_{k-1}(m_2, \ldots, m_k)$ as

$$h_m(z) = \frac{P_k + zP_{k-1}}{Q_k + zQ_{k-1}} \tag{4}$$

for $|h| = k$; the continuants are defined by the recurrence equations

$$Q_n(x_1, x_2, \ldots, x_n) = x_n Q_{n-1}(x_1, \ldots, x_{n-1}) + Q_{n-2}(x_1, \ldots, x_n),$$

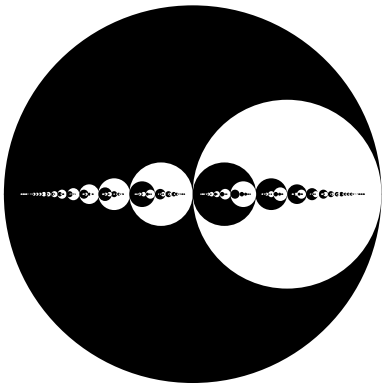


FIGURE 2. The domains $\mathcal{D}_0 \setminus \mathcal{D}_1$, $\mathcal{D}_1 \setminus \mathcal{D}_2$, $\mathcal{D}_2 \setminus \mathcal{D}_3$, $\mathcal{D}_3 \setminus \mathcal{D}_4$, $\mathcal{D}_4 \setminus \mathcal{D}_5$ represented alternatively in black and white. (The largest disk is $\mathcal{D}_0 \equiv \mathcal{D}$ which is the disk of diameter $[0,1]$.)

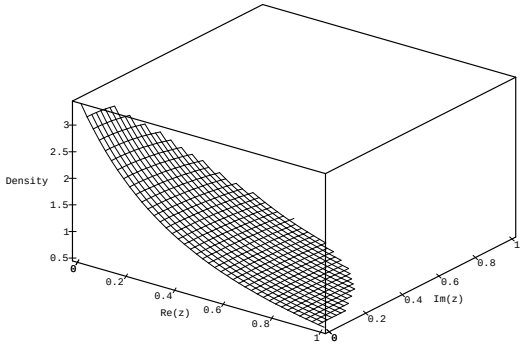


FIGURE 3. The conditional invariant density F_∞.

with $Q_0 = 1$, $Q_1(x_1) = x_1$. Then, the set of points giving more than k iterations is $[L(z) \geq k+1] = \bigcup_{|h|=k} h(\mathcal{D})$ with $h(\mathcal{D})$ the fundamental disk of diameter $h(\mathcal{I}) = [P_k/Q_k, (P_k + P_{k-1})/(Q_k + Q_{k-1})]$.

We have $\mu[h(\mathcal{D})] = \iint_{h(\mathcal{D})} f(z)dxdy = \iint_{\mathcal{D}} |h'(z)|^2 f \circ h(z)\, dxdy$, the measure μ being associated with the density f, and, remarking that the disks $h(\mathcal{D})$ are disjoint, after exchanging the sum and integral signs,

$$\varpi_k = \Pr[L \geq k+1] = \frac{1}{\mu(\mathcal{D})} \sum_{|h|=k} \mu[h(\mathcal{D})] = \iint_{\mathcal{D}} \Big(\sum_{|h|=k} |h'(z)|^2 f \circ h(z) \Big)\, dxdy. \tag{5}$$

Introducing the operator $\mathcal{H}^k_{2s}[f][z] = \sum_{|h|=k} |h'(z)|^s f \circ h(z)$, we have

THEOREM 1. *For a density f, the probability of making more than k iterations of the Gaussian algorithm is*

$$\varpi_k[f] = \frac{\iint_{\mathcal{D}} \mathcal{H}^k_4[f](z)\, dxdy}{\iint_{\mathcal{D}} f(z)\, dxdy};$$

if the density f is uniform, the probability is

$$\varpi_k[f] = \sum_{m_1,\ldots,m_k} \frac{1}{Q_k^2 (Q_k + Q_{k-1})^2},$$

and the expectation of the number of iterations is

$$\mathrm{E}[L] = \frac{5}{4} + \frac{180}{\pi^4} \sum_{d \geq 1} \frac{1}{d^2} \sum_{d < c < 2d} \frac{1}{c^2}.$$

Therefore all the objects we are studying may be expressed with $\mathcal{H}_s[f](z) = \sum_{m \geq 1} \frac{1}{|m+z|^s} f(\frac{1}{m+z})$ and its holomorphic version $\mathcal{G}_s[f](z) = \sum_{m \geq 1} \frac{1}{(m+z)^s} f(\frac{1}{m+z})$, the classical Ruelle-Mayer operator $\mathcal{G}_s$. While in the uniform case, the study of $\mathcal{G}_4$ is sufficient, in general it is necessary to study the complete family of the $\mathcal{H}_s$.

2. Properties of the Ruelle-Mayer operators and application to the analysis of the Gaussian algorithm

The Ruelle-Mayer operators $\mathcal{G}_s$ are defined on the set $A_\infty(V)$ of holomorphic functions on V, continuous on $\overline{V}$, with $V = \{|z-1| < \frac{3}{2}\}$, for s with $\Re(s) > 1$. They are nuclear operators of order 0 (very similar to infinite matrices); after transfer in another Hilbert space, they are diagonalisable with a discrete spectrum; moreover, they are Perron-Frobenius operators for $s > 1$, having a unique dominant eigenvalue $\lambda(s)$.

As a consequence, turning back to the uniform case, we have the theorem.

THEOREM 2. *The probability ϖ_k has a geometric behaviour, $\varpi_k \simeq c\lambda_4^k$, with $\lambda_4 \approx 0.1994$ and $c \approx 1.3$.*

The dynamic density $F_k(z)$ converges to a (conditional) invariant density $F_\infty(z)$ proportional to $\int_{-1}^{+1}(1-w^2)c_4(x+iyw)\,dw$, where c_4 is the dominating eigenvector of $\mathcal{G}_4$.

In the general case, we have to study generalized Ruelle-Mayer operators [3]

The spectral properties of the operator $\mathcal{H}_s$ are essentially the same as those of $\mathcal{G}_s$; there is a dominant eigenvalue $\lambda(s)$ and a dominant eigenvector which may be expressed easily in terms of $\mathcal{G}_s$. However, an interesting improvement is possible in case of functions with valuations.

THEOREM 3. *For an initial density of valuation t—$f(x,y) = |y|^t g(x,y)$, with $g \not\equiv 0$ on the real axis—the asymptotical behaviours of $\varpi_k[f]$ and $F_k[f]$ depend on the dominant spectral objects of $\mathcal{G}_{4+2t}$:*

$$\varpi_k[f] \simeq c\lambda_{4+2t}^k \tag{6}$$

and $F_\infty[f](z)$ is proportional to $|y|^t \int_{-1}^{+1}(1-w^2)^{1+t} g_{4+2t}(x+iyw)\,dw$.

3. Conclusion

These results lead to two main applications:

from Gauss to Euclid: then, we have $t \to 1$, $\lambda_{4+2t} \to \lambda_2 = 1$, and $g_{4+2t} \to g_2 = \frac{1}{\log 2}\frac{1}{1+t}$;

from Gauss to LLL: considering n vectors $b_1, \ldots, b_n$ uniformly distributed in $\mathcal{B}_n$, the unit ball of $\mathbb{R}^n$, with l_i the length of the i-th orthogonalized, the initial density has valuation $n-i-1$, and we can apply our results with use of $\mathcal{G}_{2(1+n-i)}$ [1].

We showed how to "inverse" the operator U of the Gaussian algorithm by use of a functional operator $\mathcal{G}_s$. An open question is the generalization of such a method to other algorithms.

Bibliography

[1] Daudé (H.) and Vallée (B.). – An upper bound on the average number of iterations of the LLL algorithm. *Theoretical Computer Science*, vol. 123, n° 1, 1994, pp. 95–115.

[2] Daudé (Hervé), Flajolet (Philippe), and Vallée (Brigitte). – An analysis of the Gaussian algorithm for lattice reduction. In Adleman (L.) (editor), *Algorithmic Number Theory Symposium, Lecture Notes in Computer Science*, pp. 144–158. – 1994. Proceedings of *ANTS'94*.

[3] Vallée (B.). – Le rôle des opérateurs de Ruelle-Mayer généralisés dans l'analyse en moyenne des algorithmes d'Euclide et de Gauss. – GREYC, Département d'Informatique, Université de Caen, 14032 Caen Cedex, France.

Average Case Analysis of Tree Rewriting Systems

Cyril Chabaud
LITP

June 12, 1995

[summary by Xavier Gourdon]

Abstract

A general technique is presented to easily compute the order of the average complexity of a tree rewriting system from its matrix representation. It can be used for example to prove that the average cost of the k-th differentiation is of order $n^{1+k/2}$.

1. Introduction

We aim at studying the order of the average complexity of regular tree rewriting systems. We deal with simple families of trees $\mathcal{T}$, in the sense of Meir and Moon [4]. The corresponding generating function (GF) is defined by $T(z) = z\phi(T(z))$ where $\phi(y)$ is a polynomial whose n-th coefficient is the number of constructors of arity n. For example, the GF of binary trees is defined by $T(z) = z(1 + T^2(z)) = z\phi(T(z))$ with $\phi(y) = 1 + y^2$.

Asymptotics of $T(z)$. We define $\tau > 0$ as the solution of $\tau\phi'(\tau) - \phi(\tau) = 0$ and we denote $\rho = \tau/\phi(\tau) = 1/\phi'(\tau)$. Then ρ is the dominant singularity of $T(z)$ with the Puiseux expansion

$$T(z) = \tau - \sqrt{\frac{2\phi(\tau)}{\phi''(\tau)}}\left(1 - \frac{z}{\rho}\right)^{1/2} + \sum_{n\geq 2} d_n \left(1 - \frac{z}{\rho}\right)^{n/2}.$$

From singularity analysis, we deduce the estimate of the n-th coefficient of $T(z)$

$$[z^n]T(z) = \sqrt{\frac{\phi(\tau)}{2\pi\phi''(\tau)}}\rho^{-n}n^{-3/2}\left(1 + O\left(\frac{1}{n}\right)\right).$$

An example: differentiation of trees. A typical example of a tree rewriting system is formal differentiation. We describe the action of the differentiation and copy operators on trees constructed with a binary constructor $*$ and a variable a

$$\begin{aligned} \mathrm{d}(a) &\to a & \mathrm{cp}(a) &\to a \\ \mathrm{d}(u * v) &\to \mathrm{d}(u) * \mathrm{cp}(v) + \mathrm{d}(v) * \mathrm{cp}(u) & \mathrm{cp}(u * v) &\to \mathrm{cp}(u) * \mathrm{cp}(v) \end{aligned}$$

If $B(z)$ denotes the GF of binary trees, this translates in terms of cost generating functions in the form

$$\begin{aligned} C_{\mathrm{d}}(z) &= B(z) + 2zB(z)C_{\mathrm{d}}(z) + 2zB(z)C_{\mathrm{cp}}(z), \\ C_{\mathrm{cp}}(z) &= B(z) + 2zB(z)C_{\mathrm{cp}}(z). \end{aligned}$$

Equivalently, we have the matrix representation

$$\begin{pmatrix} C_{\rm d}(z) \\ C_{\rm cp}(z) \end{pmatrix} = \begin{pmatrix} 2zB(z) & 2zB(z) \\ 0 & 2zB(z) \end{pmatrix} \begin{pmatrix} cC_{\rm d}(z) \\ C_{\rm cp}(z) \end{pmatrix} + \begin{pmatrix} B(z) \\ B(z) \end{pmatrix}. \tag{1}$$

The solution is

$$C_{\rm d}(z) = \frac{B(z)}{(1-2zB(z))^2} = \frac{B(z)(1+B^2(z))}{(1-B(z))^2(1+B(z))^2}, \quad C_{\rm cp}(z) = \frac{B(z)(1+B^2(z))}{(1-B(z))(1+B(z))}.$$

Since $B(z) = z\phi(B(z))$ with $\phi(w) = 1+w^2$, τ and ρ are easily determined: $\tau = 1$, $\rho = 1/2$.

Average complexity. The cost GF $C_{\rm cp}(z)$ writes as $F(B(z))$, where $F(w)$ is a rational function. The dominant pole of $F(w)$ is $w = \tau$ and it is simple. An application of transfer lemma of singularity analysis [2] then leads to the estimate $\overline{C_n^{\rm cp}} \sim c_1 n$ with $c_1 > 0$ for the cost of the copy operator over trees of size n. As for the cost GF $C_{\rm d}(z)$, it writes as a rational functional in $B(z)$ with the double dominant pole τ, and we deduce an average asymptotic value of the form $\overline{C_n^{\rm d}} \sim c_2 n^{3/2}$, $c_2 > 0$.

2. Regular rewriting systems

The matrix representation (1) for the cost GF's can be generalised for all regular rewriting systems [1].

THEOREM 1 (MATRIX REPRESENTATION FOR REGULAR REWRITING SYSTEMS). *The cost GF's of operators $f_1, \ldots, f_n$ of a regular system satisfy a system of the form*

$$\begin{pmatrix} C_{f_1}(z) \\ \vdots \\ C_{f_n}(z) \end{pmatrix} = M(z, T(z)) \begin{pmatrix} C_{f_1}(z) \\ \vdots \\ C_{f_n}(z) \end{pmatrix} + \begin{pmatrix} T^{r_1}(z) \\ \vdots \\ T^{r_n}(z) \end{pmatrix}, \tag{2}$$

where the r_i are the arities of the f_i's, and where the coefficient of the square matrix $M(z, T(z))$ are polynomials in z and $T(z)$ with non negative coefficients.

Thus, the expression of each cost GF is

$$C_{f_i}(z) = \frac{\det^{[i]}(\mathrm{Id} - M(z, T(z)))}{\det(\mathrm{Id} - M(z, T(z)))}, \tag{3}$$

where ${}^{[i]}A$ denotes the matrix in which the i-th column of A has been substituted by the rightmost vector of equation (2). We deduce, since $z = T(z)/\phi(T(z))$, that $C_{f_i}(z)$ writes as

$$C_{f_i}(z) = \frac{P_i(T(z))}{Q_i(T(z))},$$

where $P_i(w)$ and $Q_i(w)$ are polynomials. The average complexity of the operator f_i, defined by

$$\overline{C_n^{f_i}} = \frac{[z^n]C_{f_i}(z)}{[z^n]T^{r_i}(z)},$$

is determined by the relative position of ρ with respect to the smallest positive solution $\rho_{0,i}$ of $Q_i(T(\rho_{0,i})) = 0$ (see [1]).

THEOREM 2 (AVERAGE COST ESTIMATE). *The average cost satisfies*

(i) *If $Q_i(T(z))$ does not vanish on $(0, \rho]$, then $\overline{C_n^{f_i}} = c_1(1 + O(1/n))$;*

(ii) *if $\rho = \rho_{0,i}$, then $\overline{C_n^{f_i}} = c_2 n^{k/2}(1 + O(1/\sqrt{n}))$;*

(iii) *if* $\rho > \rho_{0,i}$, *then* $\overline{C_n^{f_i}} = c_3(\rho/\rho_{0,i})^n n^{q+1/2}(1 + O(1/\sqrt{n}))$,

with $c_j > 0$ *and* k, q *positive integers.*

In the case $\rho = \rho_{0,i}$, we have $k = s + 1$ where s is the multiplicity of the factor $(T(z) - \tau)$ in $Q_i(T(z))$.

3. Computation of the order of the average cost

It is possible to derive directly from the matrix $M(z, T(z))$ the order of the average cost of the operators of a regular system. The substance relies on Frobenius theory of matrices with nonnegative coefficients (see for instance [5]). The general technique proceeds as follows. First, decompose $M(z, T(z))$ into diagonal blocks of irreducible matrices (Definition 1), then work on each irreducible block.

3.1. Irreducible matrix case.

DEFINITION 1. A square matrix M is *irreducible* if there does not exist any permutation matrix P such that $P^{-1}MP = \left(\begin{smallmatrix} A & 0 \\ B & C \end{smallmatrix}\right)$ with A, B and C square matrices.

In other terms, an irreducible matrix is associated to a strongly connected graph. If the matrix $M(z, T(z))$ is irreducible, the order of the average complexity of the operators are easily found.

THEOREM 3. *Let* $\{f_i\}$ *be a set of operators of a regular rewriting system represented by an irreducible matrix* $M(z, T(z))$. *Then all the* $\rho_{0,i}$ *are equal to the smallest positive root* ρ_0 *of the equation* $\det(Id - M(z, T(z))) = 0$ *(take* $\rho_0 = +\infty$ *if there is no positive solution). The relative position of* ρ_0 *with respect to* ρ *is determined from the dominant eigenvalue* $r(\rho, \tau)$ *of* $M(\rho, \tau)$. *We have*

$$r(\rho, \tau) < 1 \text{ iff } \rho_0 > \rho, \qquad r(\rho, \tau) = 1 \text{ iff } \rho_0 = \rho, \qquad r(\rho, \tau) > 1 \text{ iff } \rho_0 < \rho.$$

When $r(\rho, \tau) = 1$, or equivalently $\rho_0 = \rho$, it is possible to get the exponent of n in the estimate (ii) of Theorem 1. This is the polynomial case.

THEOREM 4. *Let* $\{f_i\}$ *be a set of operators of a regular rewriting system represented by an irreducible matrix* $M(z, T(z))$. *If the dominant eigenvalue of* $M(\rho, \tau)$ *is* 1, *then the* f_i*'s have an average complexity which is linear or of order* $n^{3/2}$.

The case $n^{3/2}$ occurs only in the degenerate case where $M(z, T(z))$ does not depend on $T(z)$.

3.2. General case. In the general case, we start by finding a permutation matrix P such that $P^{-1}MP$ writes as a block diagonal matrix, each block being of the form

$$B = \begin{pmatrix} A_1 & & 0 \\ \ddots & \ddots & \\ \ddots & \ddots & A_k \end{pmatrix},$$

where each $A_i = A_i(z, T(z))$ is an irreducible square block. We also need the constraint that for all $i < j$, the submatrix of B whose lines are those of A_j and columns are those of A_i is not zero. Considering the graph represented by the matrix M, this task can be achieved thanks to Tarjan algorithm on strongly connected components (see [3, pp. 441–448] for example).

Now, each block of the form B can be considered independently. Let $C_{f_j}(z)$ be a cost GF associated to an irreducible square block A_ℓ. Expression (3) together with Theorem 3 show that the position of $\rho_{0,j}$ is the smallest positive root of $\prod_{i=1}^{\ell} \det(\mathrm{Id} - A_i)$. Thus, if ρ_i denotes the smallest positive root of $\det(\mathrm{Id} - A_i)$, for each ℓ, we need to compare ρ_ℓ with $\min_{1 \le i \le \ell - 1} \rho_i$ in order to get

the order of the average complexity of the operators f_j. In fact, Theorem 3 asserts that this task can be achieved by comparing only the dominant eigenvalues $r_i(\rho,\tau)$ of the $A_i(\rho,\tau)$'s.

In the polynomial case, the multiplicity of the factor $(T(z)-\tau)$ in the denominator of the cost GF is obtained by adding the multiplicities of this factor in the determinants $\det(\mathrm{Id}-A_i)$, yielding the exponent of n in equation (ii) of theorem 2.

3.3. Examples.

Tree shuffle. We consider binary trees $\mathcal{B}=a+o(\mathcal{B},\mathcal{B})$ and operators f and g defined on $\mathcal{B}^2$ by

$$f(a,a)\to a \quad f(o(u,v),a)\to o(u,v) \quad f(a,o(u,v))\to g(u,v)$$
$$g(a,a)\to a \quad g(o(u,v),a)\to g(u,v) \quad g(a,o(u,v))\to g(u,v)$$

$$f(o(u1,v1),o(u2,v2))\to o(f(u1,u2),f(v1,v2))$$
$$g(o(u1,v1),o(u2,v2))\to o(f(u1,u2),g(v1,v2))$$

The matrix representation of the shuffle is

$$M(z,T(z))=\begin{pmatrix}2z^2T^2(z) & 2z^2\\ z^2T^2(z) & z^2T^2(z)+2z^2\end{pmatrix}.$$

This matrix is irreducible. The eigenvalues of $M(\rho,\tau)$ are 1 and $1/4$, thus we are in the polynomial case with a linear average complexity of the operators f and g.

Formal differentiation. The classical formal double differentiation on unary-binary trees $\mathcal{T}$ with constructors `*`, `exp` and a variable has the matrix representation

$$M(z,T(z))=\begin{pmatrix}z+2zT(z) & 0 & 0\\ 2zT(z)+z & z+2zT(z) & 0\\ 2zT(z)+2z & 4zT(z)+2z & z+2zT(z)\end{pmatrix}.$$

The diagonal coefficients of $M(\rho,\tau)$ are only 1's, thus we are in the polynomial case with three blocks of irreducible matrices on the diagonal, all giving a contribution to the order of the complexity. Thus, the average complexity of the double differentiation operator is cn^2. By induction on k, it can be proved that the average cost of the k-th differentiation is of order $n^{k/2+1}$.

Bibliography

[1] Choppy (Christine), Kaplan (Stéphane), and Soria (Michèle). – Complexity analysis of term rewriting systems. *Theoretical Computer Science*, vol. 67, 1989, pp. 261–282.

[2] Flajolet (Philippe) and Odlyzko (Andrew M.). – Singularity analysis of generating functions. *SIAM Journal on Discrete Mathematics*, vol. 3, n° 2, 1990, pp. 216–240.

[3] Froidevaux (Christine), Gaudel (Marie-Claude), and Soria (Michèle). – *Types de données et algorithmes.* – McGraw–Hill, Paris, 1990.

[4] Meir (A.) and Moon (J. W.). – On the altitude of nodes in random trees. *Canadian Journal of Mathematics*, vol. 30, 1978, pp. 997–1015.

[5] Minc (Henryk). – *Nonnegative matrices.* – J. Wiley and sons, New York, 1988, *Wiley interscience series in discrete mathematics and optimization.*

Interval Algorithm for Random Number Generation

Mamoru Hoshi
Graduate School of Information Systems,
The University of Electro-Communications,
Tokyo, Japan

June 12, 1995

[summary by Vincent Dumas]

1. Introduction

This talk is based on a joint paper with Te Sun Han [1]. It presents an "interval algorithm" that solves the problem of generating a random number X with distribution $\mathbf{q} = (q_1, q_2, \ldots, q_N)$ (i.e. $\Pr[X = k] = q_k$, $1 \le k \le N$) from independent identically distributed tosses with an M-coin of distribution $\mathbf{p} = (p_1, p_2, \ldots, p_M)$. This problem was set by Roche [2] (variants of this problem were studied by von Neumann, Elias, Knuth and Yao). The efficiency of the algorithm is measured by L^*, which is the expected number of tosses required to generate X. Roche proved that the optimal algorithm should satisfy:

$$\frac{H(\mathbf{q})}{H(\mathbf{p})} \le L^* \le \frac{H(\mathbf{q}) + f(\mathbf{p})}{H(\mathbf{p})},$$

where H is the entropy function (see Appendix) and

$$f(\mathbf{p}) = \ln(e/p_{\min}), \qquad \text{where} \qquad p_{\min} = \min_{1 \le j \le M} p_j.$$

The upper bound is satisfied by a probabilistic algorithm.

Han and Hoshi propose an "interval algorithm" that satisfies the upper bound with

$$f(\mathbf{p}) = \ln[2(M-1)] + \frac{h(p_{\max})}{1 - p_{\max}}, \qquad \text{where} \qquad p_{\max} = \max_{1 \le j \le M} p_j,$$

with $h(p) = -p \ln p - (1-p)\ln(1-p)$. No choice of function f seems to be essentially better than any other one. The assumed superiority of the interval algorithm is that it is *deterministic* and *easy to implement.*

2. Interval algorithm

Let $\mathbf{p}$ be the original distribution. Let us fix a partition of $[0, 1)$ according to $\mathbf{p}$, that is a sequence

$$\alpha_0 = 0 < \alpha_1 < \cdots < \alpha_M = 1,$$

such that $\alpha_j - \alpha_{j-1} = p_j$ for all j. Now any interval $[a, b)$ may be partitioned into the subintervals $I_j([a, b))$, $1 \le j \le M$, with

$$I_j([a, b)) = [a + (b-a)\alpha_{j-1}, a + (b-a)\alpha_j).$$

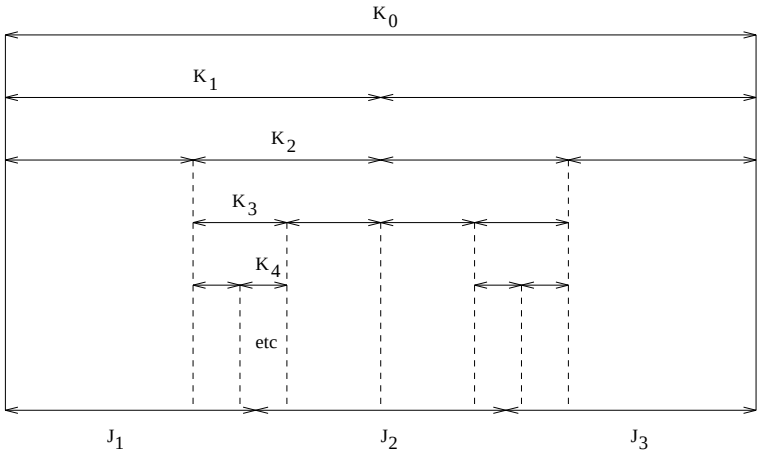


FIGURE 1. Example of sequence (K_n) ($\mathbf{p} = (1/2, 1/2)$, $\mathbf{q} = (1/3, 1/3, 1/3)$).

Let $\mathbf{q}$ be the distribution we want to generate. Fix a partition

$$\beta_0 = 0 < \beta_1 < \cdots < \beta_N = 1$$

of $[0, 1)$ according to $\mathbf{q}$ ($\beta_k - \beta_{k-1} = q_k$), and set $J_k = [\beta_{k-1}, \beta_k)$.

The interval algorithm is defined as follows:

(1) set $n = 0$ and $K_0 = [0, 1)$;
(2) if $K_n \subset J_k$ for some k, then stop the algorithm and set $X = k$;
(3) else flip the M-coin (with probability distribution $\mathbf{p}$). The result is a number $M_n \in \{1, \ldots, M\}$. Set $K_{n+1} = I_{M_n}(K_n)$ and go to (2).

This procedure is illustrated in Figure 1.

With probability one this algorithm terminates in finite time, and generates a random number X, which is a deterministic function of $Y = K_\infty$. Let $\mathcal{Y}$ be the set of all possible values of Y. By construction, $\mathcal{Y}$ is a partition of $[0, 1)$, and any $y \in \mathcal{Y}$ may be obtained with probability $|y|$ (where $|y|$ denotes the length of interval y). In consequence, we fall in J_k with probability $|J_k| = q_k$, which means that X has distribution $\mathbf{q}$ as expected.

Now denote by L the number of tosses necessary to get X. From basic results on entropy in tree algorithms, we get that

$$L^* = \mathrm{E}(L) = \frac{H(Y)}{H(\mathbf{p})}.$$

Moreover, since X is a (deterministic) function of Y, then $H(Y) \geq H(X) = H(\mathbf{q})$, which yields

$$L^* \geq \frac{H(\mathbf{q})}{H(\mathbf{p})}.$$

3. Upper bound

In order to get an upper bound on $H(Y)$ (and then on L^*), the authors introduce a new variable W, such that

(1) W is a function of Y;
(2) W has $2(M-1)$ possible values;

(3) conditionally on (W, X) being equal to some (w, k), we have

$$Y \succ \mathrm{Geom}(p_{\max}),$$

where $\mathrm{Geom}(p)$ denotes the geometric distribution of parameter p:

$$\Pr[\mathrm{Geom}(p) = i] = (1-p)p^i.$$

Then we will get that: $H(Y) = H(Y, W, X) = H(X) + H(W|X) + H(Y|(W, X))$, with

$$H(X) = H(\mathbf{q}), \quad H(W|X) \leq \ln[2(M-1)], \quad H(Y|(W,X)) \leq H(\mathrm{Geom}(p_{\max})) = \frac{h(p_{\max})}{1-p_{\max}},$$

which yields the announced bound.

In order to define W, set $X = k$ and consider the possible values of Y, that is all the intervals $y \in \mathcal{Y}$ such that $y \subset J_k$. We may organize them as follows. There is a unique sequence of tosses (M_n) such that, for all n, $K_n = [\gamma, \delta)$ with $\gamma \leq \beta_{k-1}$ and $\delta > \beta_{k-1}$ (resp. with. $\gamma < \beta_k$ and $\delta \geq \beta_k$): this is the *upward* sequence (resp. the *downward* sequence) associated to J_k; it is finite only if $\gamma = \beta_{k-1}$ (resp. if $\delta = \beta_k$) for some K_n. Now any possible value of Y corresponds to a unique, finite sequence of tosses $(M_n(y))_{0 \leq n \leq n(y)}$, and we can check that

$$M_n(y) = M_n, \qquad 0 \leq n < n(y)$$

is valid for (M_n) equal to either the upward sequence or the downward sequence.

For a given y, set $\mathrm{sign}(y) = $ upward (resp. $\mathrm{sign}(y) = $ downward) if y derives from an upward sequence (resp. a downward sequence), and $M(y) = M_{n(y)}(y)$ (the value of the last toss that stops the algorithm at y). One can check that if $\mathrm{sign}(y) = $ upward (resp. if $\mathrm{sign}(y) = $ downward), then $M(y)$ cannot be equal to 1 (resp. $M(y)$ cannot be equal to M); in consequence, there are only $2(M-1)$ possible values for $(\mathrm{sign}(y), M(y))$. We may now define the new random variable $W = (\mathrm{sign}(Y), M(Y))$ which obviously satisfies properties (1) and (2). Moreover, if $X = k$ and $W = (s, m)$, then all the possible values of Y derive from the same upward or downward sequence (M_n), and they may be ordered in a sequence (y_l) such that $n(y_l)$ is strictly increasing. In consequence, the interval algorithm yields y_l with probability

$$p(y_l) = \left(\prod_{n=0}^{n(y_l)-1} p_{M_n} \right) p_m,$$

which implies that $p(y_l) \leq p_{\max} p(y_{l-1})$: property (3) may be deduced from this inequality.

4. Conclusion

The interval algorithm may be adapted to generate the first n terms of a finite state space Markov chain; the average cost L^*/n is then asymptotically optimal. Independent identically distributed tosses with an M-coin may also be replaced by a Markov chain.

Appendix: basic properties of the entropy function

The entropy of a distribution $\mathbf{a} = (a_i)_{i \in I}$ (where I is countable) is defined by:

$$H(\mathbf{a}) = -\sum_{i \in I} a_i \ln a_i.$$

The notation $H(A)$ is also used if A is a random variable with distribution $\mathbf{a}$. If $\mathrm{Card}(I) = P$, then $H(A) = H(\mathbf{a}) \leq \ln P$.

Since a pair of random variables (A, B) is a random variable, one may also consider the entropy $H(A, B)$. If $B = f(A)$ (where f is deterministic), then $H(A) \geq H(B)$ (notice that it implies $H(A, B) = H(A)$).

In the general case, denote by $A/B = b$ the distribution of A conditioned on $B = b$ (it is assumed that $\Pr(B = b) > 0$). Set $f(b) = H(A/B = b)$. Then one may define

$$H(A|B) = \mathrm{E}[f(B)],$$

which satisfies: $H(A|B) = H(A, B) - H(B)$.

Now, consider two distributions $\mathbf{a} = (a_i)_{i \geq 1}$ and $\mathbf{b} = (b_i)_{i \geq 1}$ ordered in decreasing probabilities ($a_i \geq a_{i+1}$ and $b_i \geq b_{i+1}$, for all i). The partial ordering $\mathbf{a} \succ \mathbf{b}$ is defined by:

$$\sum_{i=1}^{j} a_i \geq \sum_{i=1}^{j} b_i, \qquad \forall j \geq 1.$$

If $\mathbf{a} \succ \mathbf{b}$, then $H(\mathbf{a}) \leq H(\mathbf{b})$ (this is indeed valid for all the concave, symmetric functions).

Bibliography

[1] Han (Te Su) and Hoshi (Mamoru). – Interval algorithm for random number generation. – May 1995. Preprint.

[2] Roche (J. R.). – *Efficient generation of random variables from biased coins.* – Bell Technical Report n° 20878, AT&T Laboratories, 1992.

Algorithmic Problems in Non-Cabled Networks

Philippe Jacquet
INRIA Rocquencourt

February 6, 1995

[summary by Xavier Gourdon]

Abstract

Due to the specific nature of radio networks, the channel access in radio local area networks (LAN) is different from cabled LAN. The HIPERLAN standard for radio networks will provide a 24Mbps data rate transmission. It has a special feature, called active signalling, which can be used to provide an efficient channel access mechanism.

1. Active signalling

The channel access in radio LANs has to face special problems. Unlike wired LANs, nodes cannot build a complete history of the network from the fragments of feedback they obtain from the channel. This feature makes the collision detection techniques in radio networks different from cabled networks.

The active signalling feature of the radio LAN standard HIPERLAN provides an efficient channel access mechanism. It consists in requiring each node that wants to access the channel to send a certain sequence of on/off's as a preamble to each packet transmission. This sequence is encoded according to a random pattern whose details will be described later. The objective is to use these patterns to select (with a high probability) only one node so that no collision occurs during packet transmission. The patterns are also functions of the access priority assigned to the packet. During the transmission of its pattern and when it is in the "off" period, the node senses the channel: if it detects any other signal, then the node stops its pattern transmission and defers until the next attempt.

2. HIPERLAN active signalling pattern selection

The HIPERLAN active signalling pattern is divided into two consecutive phases, the access priority assertion phase and the contention phase.

2.1. The access priority assertion phase. The first slots of the pattern are dedicated to priority signalling.

The priority phase consists in leaving a certain number of idle (off) slots before one busy (on) slot. This number of slots is equal to 5 (maximum number of priority levels) minus the priority level. The priority assertion phase ends with the first busy slot encountered, called the priority pulse. Therefore, only the contenders with the highest access priority level survive to the priority assertion phase. Figure 1 shows an example where node B, on access priority level 3, beats node A on access priority level 2.

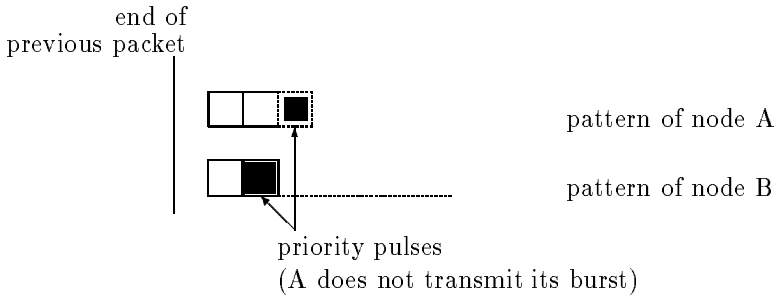


FIGURE 1. How to win priority contention

2.2. The contention phase. The contention phase is divided into two consecutive parts, the elimination part and the tail selection part. The role of the elimination part is to select a small number of survivors from a large number of contenders. The tail selection part tries to select only one survivor from a small number of contenders with a high probability. If more than one survivor is selected at the end of the contention phase, a collision occurs.

The elimination phase. It consists in enlarging the priority pulse with a random number of slots. Each node stretches its pulse independently of the other nodes and according to a geometric distribution of probability $p = 1/2$. Therefore the pulse is larger than k slots with probability $1/2^k$.

After the stretched pulses, the node leaves an idle slot, called the *survival verification slot* where it senses the channel. Only the contenders which simultaneously hold the highest access priority level and select the longest stretched pulse survive to the elimination part. Figure 2 shows an example where node A with stretch length of 1 slot is eliminated by node B with stretch length of 2 slots.

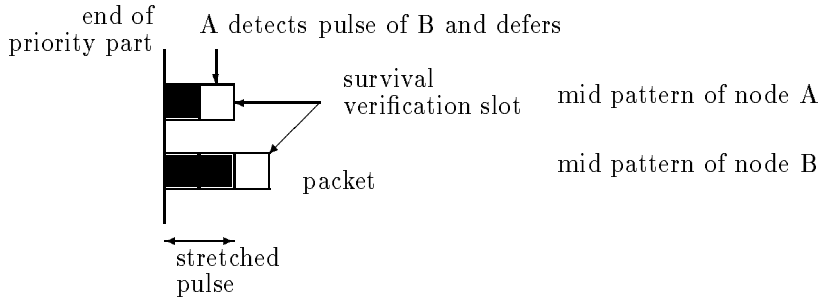


FIGURE 2. How to win the elimination part

The tail selection phase. This phase is also called the "yield" part and follows just after the survival verification slot. The nodes which survived to the elimination part again select a random number of slots according to a geometric rate $1 - r$ with $r = 1/8$. But instead of transmitting busy slots again as with the stretched pulse procedure, the contenders terminate their pattern with a number of idle slots equal to their respective new selected numbers. Thus if a node detects no signal during its silent period, then the node transmits its packet. Otherwise, the node defers until the next access cycle.

Therefore, only the nodes whose patterns simultaneously present the highest access priority level, the longest stretched pulse and the shortest "yield" part gain the right to transmit their packet.

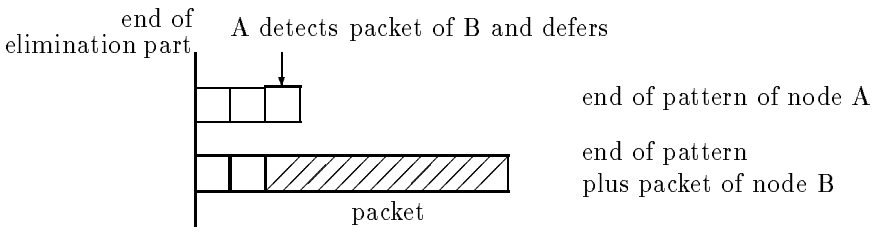


FIGURE 3. How to win the tail selection part

Figure 3 shows an example where node B with yield length of 2 slots is selected before node A with yield length of 3 slots.

3. Performance evaluation of HIPERLAN active signalling channel access

The contention phase has a certain length, called overhead. Starting with n contenders at the same highest priority level, we prove that the average contention overhead is $\log_2 n + O(1)$. Also, whatever the number of contenders, the residual collision rate on packet transmission is less than 3.5%.

3.1. Analytic evaluation of the elimination phase. Given n contenders at the highest priority level, we denote by S_n the average number of survivors after the elimination phase, L_n the average stretched pulse length of these survivors, and p_n the probability of having a single survivor.

With p the geometric stretching probability and $q = 1 - p$, the following recursions hold:

$$S_n = nq^n + \sum_{k=1}^{n} \binom{n}{k} p^k q^{n-k} S_k,$$

$$p_n = \sum_{k=1}^{n} \binom{n}{k} p^k q^{n-k} p_k,$$

$$L_n = 1 - q^n + \sum_{k=1}^{n} \binom{n}{k} p^k q^{n-k} L_k.$$

Referring to the general methodology used in the analysis of algorithms (translating in terms of generating functions, then using Mellin transforms to estimate harmonic sums, see [1] for example) leads to the following result.

THEOREM 1. *Asymptotically,*

$$S_n = \frac{q}{p \log(1/p)} + P_1\left(\frac{\log n}{\log(1/p)}\right) + O(1/n),$$

$$p_n = \frac{q}{\log(1/p)} + P_2\left(\frac{\log n}{\log(1/p)}\right) + O(1/n),$$

$$L_n = \frac{\log n}{\log(1/p)} + \frac{\gamma}{\log(1/p)} - \frac{1}{2} + P_3\left(\frac{\log n}{\log(1/p)}\right) + O(1/n).$$

The $P_i(x)$'s are 1-periodic functions with amplitude of order $\exp(-\pi^2/\log(1/p))$, and γ is the Euler constant.

For $p = 1/2$, the $P_i(x)$'s have amplitude less than 10^{-5}, and $S_n \approx 1.44$, $p_n \approx 0.72$, $L_n \approx \log_2 n + 0.33$. These approximate values are quite good for $n \geq 20$. As expected, they show that the elimination phase selects a small number of survivors from a large number of contenders.

3.2. Analytic evaluation of the entire contention phase. This time, S_n, L_n and p_n denote the same quantities as before but taken at the end of the entire contention phase. With $1 - r$ the geometric rate in the yield part, the entire contention phase leads to the following recursions:

$$S_n = \frac{nr}{1-(1-r)^n} q^n + \sum_{k=1}^{n} \binom{n}{k} p^k q^{n-k} S_k,$$

$$p_n = \frac{nr(1-r)^{n-1}}{1-(1-r)^n} q^n + \sum_{k=1}^{n} \binom{n}{k} p^k q^{n-k} p_k,$$

$$L_n = 1 + \frac{(1-r)^n}{1-(1-r)^n} q^n + \sum_{k=1}^{n} \binom{n}{k} p^k q^{n-k} L_k.$$

Notice that the quantities $\frac{nr}{1-(1-r)^n}$, $\frac{nr(1-r)^{n-1}}{1-(1-r)^n}$ and $\frac{(1-r)^n}{1-(1-r)^n}$ are the average number of allowed transmissions, the probability of having only one transmission and the average number of slots before first transmission in a yield contention involving n contenders, respectively.

THEOREM 2. *Asymptotically,*

$$S_n = \frac{q}{\log(1/p)} \sum_{k \geq 0} \frac{r(1-r)^k}{1-(1-r)^k q} + P_1\left(\frac{\log n}{\log(1/p)}\right) + O(1/n),$$

$$p_n = \frac{q}{\log(1/p)} \sum_{k \geq 0} \frac{r(1-r)^k}{1-(1-r)^{k+1} q} + P_2\left(\frac{\log n}{\log(1/p)}\right) + O(1/n),$$

$$L_n = \frac{\log n}{\log(1/p)} + \frac{\gamma}{\log(1/p)} - \frac{1}{2} - \frac{\log\left(\prod_{k\geq 0}(1-(1-r)^k q)\right)}{\log(1/p)} + P_3\left(\frac{\log n}{\log(1/p)}\right) + O(1/n).$$

The $P_i(x)$'s are 1-periodic functions with amplitude of order $\exp(-\pi^2/\log(1/p))$.

For the value $p = 1/2$ and $r = 1/8$, we obtain the leading terms in S_n, p_n and L_n equal to $1.0302\ldots$, $0.9713\ldots$, and $\log_2 n + 7.1393\ldots$ respectively.

3.3. Network performance analysis. As expected, there is a high probability that the contention phase selects only one survivor. The size of the overhead is $\log_2 n + O(1)$ where n is the number of contenders with the highest priority level. The throughput Thr_n is defined by

$$\mathrm{Thr}_n = \frac{p_n \mathcal{L}}{L_n + \mathcal{L} + 1},$$

where $\mathcal{L}$ is the average packet size. For n not too large (say $n \leq 32$) and for a typical value of $\mathcal{L} = 40$, the throughput is relatively stable at a value close to 0.8. This outlines the important benefit obtained from active signalling access schemes over pure carrier sense schemes (CSMA), as used by Ethernet: the throughput in CSMA rapidly collapses to 0.

Bibliography

[1] Flajolet (P.) and Sedgewick (R.). – Digital search trees revisited. *SIAM Journal on Computing*, vol. 15, n° 3, August 1986, pp. 748–767.

Minimal 2-dimensional Periodicities and Maximal Space Coverings

Mireille Régnier
INRIA Rocquencourt

May 29, 1995

[summary by Mireille Régnier]

1. Introduction

String searching can be generalized to "multidimensional search" or "multidimensional pattern matching": a multidimensional pattern, p, most often an array and usually connected and convex, is searched in a multidimensional array, the text, t. A strong interest appeared recently [3, 2, 4]. Notably, the duel paradigm improves average and worst-case complexity of pattern matching. Knowing for each position of self-overlap a mismatching position—the *witness*—allows to eliminate one of the two candidates by one question—the *duel*. One studies here pattern periodicities and space coverings. One proposes a period definition valid in any dimension and consistent with the more general definitions in dimension 1, i.e. on words. We prove here that a periodic pattern is generated by a subpattern, and the subpattern, as well as the generating law and the link to the regular distribution of periods, is exhibited. The exceptions to this regularity , the degenerated periods, are interpreted as "border effects". They derive from some regularity of the generating subpattern, a basic phenomenon in dimension 1. This allows for a classification of periodicities valid in any dimension, and detailed in dimension 2. Notably, the number of periodicity classes appear linear in the dimension. Also, one provides a full characterization of sources positions, including the degenerated ones that are essential to the design and correctness of 2D pattern matching algorithms. This considerably refines and achieves the previous classification by [1], and even the extended results in [4], and allows for a classification of space coverings, where non-degenerated periodicities appear essential. One exhibits relationship between the periods of a pattern and the possible space coverings by the same pattern. This is relevant both to the derivation of the theoretical complexity of d-dimensional pattern matching and to algorithmic issues.

The simple remark that the set of invariance vectors almost has a monoïd structure provides the link to the well studied periodic functions in $\mathbb{Z}^d$. Using their properties leads to a great simplification of the proof of previous results in the area. Additionally, it provides tools for a generalization to any dimension. Finally, the paper provides knowledge to derive efficient pattern preprocessing. In particular, the characterization of minimal generating sub-patterns reduces (partially) periodicity and witness computation to well known problems on words. This allows for using the large toolkit of 1D algorithms to determine periodicities. A preliminary version of this work appeared in [6].

2. Formalism

Basic Notations. A d-dimensional pattern p is a d-dimensional array whose values range on some alphabet A. Given a vector $\vec{u}$, we denote $\vec{u}[i]$ or $\vec{u}_i$ its i-th coordinate. Let P be the set of vectors $\vec{u}$ such that $|\vec{u}[i]| \leq l_i$ where l_i is some integer, called the i-th dimension of p.

B	G	c	d	e	f	g	h	a	*b*	c	d	e	f	g	h	a
C	D	k	l	m	n	i	j	k	l	m	n	i	j	k	l	m
g	h	*a*	b	c	d	e	f	g	h	a	*b*	c	d	e	f	g
i	j	k	l	m	n	i	j	k	l	m	n	i	j	k	l	m
e	f	g	h	**a**	b	c	d	e	f	g	h	a	*b*	c	d	e
i	j	k	l	m	n	i	j	k	l	m	n	i	j	*k*	l	m
c	d	e	f	g	h	**a**	b	c	d	e	f	g	h	a	X	c
i	j	k	l	m	n	i	j	k	l	m	n	i	j	k	l	m
a	b	c	d	e	f	g	h	*a*	b	c	d	e	E	g	G	X

FIGURE 1. Radiant biperiodic pattern

DEFINITION 1. Two vectors $\vec{u}$ and $\vec{v}$ are said in the same direction if and only if, for any i: $\vec{u}_i.\vec{v}_i \geq 0$. A vector $\vec{u}$ *dominates* a vector $\vec{v}$ in the same direction if and only if, for any i, $|\vec{v}_i| \leq |\vec{u}_i|$. A vector $\vec{u}$ is minimal if it does not dominate any vector.

We are interested in shifts such that the two copies are consistent in the overlapping area.

DEFINITION 2. A vector $\vec{u}$ is an *invariance* vector for p if and only if, for any $\vec{v} \in P$, one has $p[\vec{v}+\vec{u}] = p[\vec{v}]$. A couple $(\vec{u},\vec{v})$ of invariance vectors is said an *invariance couple* if and only if $\forall j : \sum_j |u_j| + |v_j| \leq l_j$. It is simple if $\vec{u}$ and $\vec{v}$ are collinear. These invariance vectors are said *simple*. We note I the set of invariance vectors.

3. Main results

Lattice distribution of invariance vectors. If a pattern p admits a non-simple invariance couple, it is said *biperiodic* (see Figure 1). We have:

DEFINITION 3. Given a lattice L with basis $(\vec{u},\vec{v})$, we denote $FC_{\vec{u},\vec{v}} = \{\lambda\vec{u}+\mu\vec{v}; 0 \leq \lambda,\mu < 1\}$. A S-path is a chain $\vec{w}_1 \ldots \vec{w}_k$ of vectors in p such that, for any i, either $\vec{w}_{i+1} - \vec{w}_i$ or $\vec{w}_i - \vec{w}_{i+1}$ is in S. Given two vectors $(\vec{u},\vec{v})$, the *free zone* $FZ_{\vec{u},\vec{v}}$ is the set of points $\vec{w}$ in p such that there exists no $(\vec{u},\vec{v})$-path interior to P to $FC_{\vec{u},\vec{v}}$. The *periodicity domain* is $p - FZ_{\vec{u},\vec{v}}$. The *border* is:

$$B = p - \cup_{\vec{x},\vec{y}\in L^2}\{\vec{w} \mid (\vec{w}+\vec{x},\vec{w}+\vec{y}) \in p^2, dir(\vec{w}) = dir(\vec{x}) = dir(\vec{y})\}.$$

THEOREM 1. *Let p be biperiodic. For any invariance couple $(\vec{u},\vec{v})$ exists a lattice L such that:*

$$I \subseteq L \cup B \cup FZ_{\vec{u},\vec{v}}, \tag{1}$$

where B is the border of L. If L admits two simple vectors, then (1) reduces to:

$$I \subseteq L \cup B. \tag{2}$$

$(\vec{u},\vec{v})$ is said a non-degenerated invariance couple *and L is said a non-degenerated lattice. A pattern admits at most one non-degenerated lattice, called the* canonical lattice *and denoted $L_{\vec{E},\vec{F}}$ where $(\vec{E},\vec{F})$ is a basis. The invariance vectors in $\tilde{I} = I - B_{\vec{E},\vec{F}}$ are named the non-degenerated invariance vectors. p is said a non-degenerated biperiodic pattern.*

Figure 1 provides an example where $\vec{E} = [4,4]$ and $\vec{F} = [6,2]$. It is worth noticing that a basis is not necessarily made of invariance vectors: this is intrinsically 2D. Similar phenomena occur on any set of collinear vectors: e.g. a regular distribution of invariance vectors and a degeneracy paradigm.

Periodicity classification. A pattern is:

(1) *non-periodic:* no invariance couple
(2) *monoperiodic:* exists one simple invariance couple; all invariance couples are simple.
(3) *biperiodic:* exist one non-simple invariance couple. If the associated lattice is non-degenerated, the pattern is said non-degenerated biperiodic. It divides into two subclasses:
   (a) *fundamental biperiodic* or *lattice periodic:* all lattice vectors are invariance vectors.
   (b) *non fundamental biperiodic* or *radiant periodic:* all invariant lattice vectors are in the same direction.

Word properties. It appears from our example kind of a word repetition. In 1D, minimal generators and periods are based on primality notion on words. Extending this *primality* notion to dimension 2. provides an alternative point of view to the characterization of I as a subset of a lattice $L_{(\vec{E},\vec{F})}$ plus its border $B_{(\vec{E},\vec{F})}$. As a major algorithmic consequence, it allows for using 1D algorithms to search for periodicities, hence witnesses. Also, it simplifies the proofs [7].

DEFINITION 4. Let p be a non-degenerated biperiodic pattern. Let $(\vec{E},\vec{F})$ be a fundamental basis such that a fundamental lattice cell $FC_{\vec{E},\vec{F}}$ is in the periodicity domain. Denote i its direction, and j the other direction. Let $\delta = GCD(\vec{E}_j,\vec{F}_j)$ and $L = \inf\{k \geq 0; k\vec{e_i} \in L_{\vec{E},\vec{F}}\}$. Define for any (λ,μ) in $[0\ldots L-1]\times[0\ldots\delta-1]$, $\vec{w}_{\lambda,\mu}$ as the only vector in $FC_{\vec{E},\vec{F}}$ such that:

$$\vec{w}_{\lambda,\mu} - (\lambda\vec{e_i} + \mu\vec{e_j}) \in L_{\vec{E},\vec{F}}.$$

Let $p_{\lambda,\mu}$ be $p[\vec{w}_{\lambda,\mu}]$; let s_μ be the primitive word associated to the word $p_{0,\mu-1}\ldots p_{L-1,\mu-1}$. The sequence $(s_\mu)_{1\leq\mu\leq\delta}$ is the *linear canonical generator* in direction i.

Remark that the existence and uniqueness of $\vec{w}_{\lambda,\mu}$ is a direct consequence of Euclid's theorem and that $(\vec{E},\vec{F})$-periodicity implies that (s_i) is independent of the fundamental basis chosen.

THEOREM 2. *Let p be a non-degenerated biperiodic pattern, and $(s_i)_{1\leq i<\delta}$ be the associated linear generator. Then, any vector $\vec{w}$ in the periodicity domain and in the same direction satisfies:*

$$p[\vec{w}] = s_\mu(\lambda \bmod |s_\mu|) \tag{3}$$

where (λ,μ) is defined by the equation $\vec{w} - \vec{w}_{\lambda,\mu} \in L_{\vec{E},\vec{F}}$. One has $L = GCM(|s_\mu|) = \frac{|FC_{\vec{E},\vec{F}}|}{\delta}$.

Intuitively, a biperiodic pattern p is made of δ patterns that repeat indefinitely, except maybe for the borders: rows (or columns) i, $i \in \{1\ldots\delta\}$ are linear concatenations of strings s_i^* and row $j+\delta$ is equal to row j shifted by some value α. In Figure 1, we have $\delta_1 = \delta_2 = 2$ and $s_1 = abcdefgh$ and $s_2 = ijklmn$.

Position of sources. . We remark that (3) holds for any $\vec{w}$ if p is fundamental biperiodic. We show that if a vector $\vec{w}$ in $L_{\vec{E},\vec{F}} \cap T$ is not an invariance vector then $P_{\vec{w}}$ contains a point that *violates* $(\vec{E},\vec{F})$ *periodicity*: capital characters in Figure 1. Extremal such points, $[15,2],[16,0]$ and $[1,8]$, lead to the exclusion of $[8,0],[6,2]$ and $[0,8]$ from I (represented by bolded **a**).

Maximal Coverings Classification. One proves that two copies of p shifted by $\vec{u}$ and $\vec{v}$ are mutually consistent if and only if $\vec{u}-\vec{v}$ is an invariance vector or $p_{\vec{u}} \cap p_{\vec{v}} = \emptyset$. One defines a $(\vec{u},\vec{v})$-lattice covering as a set of interleaved $\vec{u}$-overlapping sequences where two neighbouring sequences are shifted by $\vec{v}$. It is *regular* if $\vec{u}+\vec{v} \in T$, else it is said *extended*. It steadily follows:

THEOREM 3. *A maximal covering of the 2-dimensional space by a pattern p is either of the three:*

(1) *tiling,*
(2) *a tiling of $\vec{u}$-overlapping sequences where $\vec{u}$ is a minimal invariance vector.*
(3) *a $(\vec{u}, \vec{v})$-lattice coverings. It is* regular *and $(\vec{u}, \vec{v})$ is a basis of the canonical lattice if p is biperiodic; otherwise it is extended.*

Remark that extended lattice coverings are an extension of the covering notion, where some "holes" appear in the representation. This is pertinent for algorithmic issues as it allows to determine the *maximum* number of occurrences of a given pattern, a parameter related to the worst-case complexity.

4. Hints for the proofs

One shows that the sum of invariance vectors is an invariance vector *almost everywhere*, and characterizes this zone of non-invariance, the free zone, that creates "border effects". This additive property allows to use general results on biperiodic functions on Z^2 and prove a lattice distribution of almost all invariance vectors. Notice this vectorial approach provides a very short proof of the previous results in [1, 4]. Many proofs rely on the Factorisation Theorem [5]: equation $ab = ba$ implies that a and b are powers of a same primitive word. For example, in Theorem 2, equation (3) implies that, for any μ, $|s_\mu|$ divides L. Otherwise, for some j, one has $L \bmod |s_j| = \alpha \neq 0$. With $a = s_j[1] \ldots s_j[\alpha]$ and $b = s_j[\alpha+1] \ldots s_j[|s_j|]$, s_j factors as $s_j = ab = ba$ which contradicts the primitivity. Hence, $GCM(|s_j|)$ divides L. Also, (3) implies that $GCM(|s_j|)\vec{e_i}$ is a lattice vector, hence $L\vec{e_i}$, by the minimality property.

A major consequence of these word properties is the possibility to compute the linear generator, hence the fundamental basis, from any fundamental parallelogram. One initially computes δ as $GCD(\vec{u}_j, \vec{v}_j)$ and L as $\inf\{k; k\vec{e_i} \in L_{\vec{u},\vec{v}}\}$. For each of the δ sequences $p_{\lambda,\mu}$ defined, one can extract the associated primitive word s_j. One may use the well known 1D algorithm that searches for the primitive seed of a word (for instance the preprocessing of Knuth-Morris-Pratt). Then, one can compute all witnesses between two sequences s_j and s_k. This determines whether the set is cyclic (not minimal) and $(\vec{E}, \vec{F})$ steadily follows. An implementation and other applications are described in [7].

Bibliography

[1] Amir (A.) and Benson (G.). – Two-dimensional periodicity and its application. In *SODA'92*. – 1992. Proceedings of the 3rd Symposium on Discrete Algorithms, Orlando, FL.

[2] Amir (A.), Benson (G.), and Farach (M.). – Alphabet independent two dimensional matching. In *STOC'92*, pp. 59–67. – 1992. Victoria, BC.

[3] Baeza-Yates (R.) and Régnier (M.). – Fast algorithms for two dimensional and multiple pattern matching. *Information Processing Letters*, vol. 45, n° 1, 1993, pp. 51–57.

[4] Galil (Z.) and Park (K.). – Truly alphabet independent two-dimensional pattern matching. In *FOCS'92*. – IEEE, 1992. Proceedings of the 33rd IEEE Conference on Foundations of Computer Science, Pittsburgh, USA.

[5] Lothaire (M.). – *Combinatorics on Words*. – Addison-Wesley, 1983, *Encyclopedia of Mathematics and its Applications*, vol. 17.

[6] Régnier (M.) and Rostami (L.). – A unifying look at d-dimensional periodicities and space coverings. In *CPM'93. Lecture Notes in Computer Science*, vol. 684, pp. 215–227. – Springer-Verlag, 1993. Proceedings of the 4th Symposium on Combinatorial Pattern Matching, Padova, Italy.

[7] Régnier (M.) and Rostami (L.). – Minimal d-dimensional and Maximal Space Coverings, 1995.

Reversing a Finite Sequence

Loïc Pottier
INRIA Sophia Antipolis

March 6, 1995

[summary by Philippe Dumas]

The concept of reversing a finite sequence is best introduced by an example. Define a sequence of vectors x_i by the formula $x_i = f_i(x_{i-1})$ for $i = 1, \ldots, p+1$; here x_0 and the functions f_i's are given. More precisely the functions $f_1, \ldots, f_p$ map $\mathbb{R}^m$ into $\mathbb{R}^m$ and the last one f_{p+1} maps $\mathbb{R}^m$ into $\mathbb{R}$. For each i, the variable x_i is a function of x_0, $x_i = g_i(x_0)$. Moreover let us assume that all these functions are differentiable. We want to compute the Jacobian matrix $J_{g_{p+1}}(x_0)$, which expresses the partial derivatives of x_{p+1} with respect to the components of x_0. By the chain rule, this matrix is expressed as a product of matrices,

$$J_{g_{p+1}}(x_0) = J_{f_{p+1}}(x_p) \times J_{f_p}(x_{p-1}) \times \cdots \times J_{f_1}(x_0).$$

The matrix $J_{g_{p+1}}(x_0)$ is a row matrix of type $1 \times m$, while the matrices in the product are square matrices of type $m \times m$ except the leftmost one, which is a row matrix of type $1 \times m$. The first idea which comes to mind is the following. We compute $J_{f_1}(x_0)$ and we store it; next we compute x_1, the Jacobian matrix $J_{f_2}(x_1)$ and the product $J_{g_2}(x_0) = J_{f_2}(x_1) \times J_{f_1}(x_0)$; we store this product, we compute x_2, the matrix $J_{f_3}(x_2)$, the product $J_{f_3}(x_2) \times J_{g_2}(x_0)$ and so on. At each step of the computation, we store a $m \times m$ matrix. If m is large (a value of about 10^6 is possible), this method is not practical. So, we apply another strategy. We first compute x_p and the Jacobian matrix $J_{f_{p+1}}(x_p)$; we store it; next we compute x_{p-1} and the Jacobian matrix $J_{f_p}(x_{p-1})$; we compute the product $J_{f_{p+1}}(x_p) \times J_{f_p}(x_{p-1})$ and we store it, and so on. The gain of storage is evident: each time we store a $1 \times m$ matrix in place of a $m \times m$ matrix. But there is a waste of time because we compute again and again the values $x_1, \ldots, x_p$. Obviously, we could store these values but the available memory has a limited size.

The problem of reversing the sequence $x_0, x_1, \ldots, x_p$ may be now formulated. We want an algorithm which provides the values $x_p, x_{p-1}, \ldots, x_1$ in this order and costs the minimal amount of time, knowing that each computation $x_i = f_i(x_{i-1})$ takes one unit of time and only r values may be stored at a time. Such an algorithm provides the value x_i only by computation from the previous value x_{i-1} or by retrieval from memory. Several authors have addressed this problem. Baur and Strassen [2] used the idea we presented as an introduction to study the complexity of partial derivatives. Abbot and Galligo [1] gave an optimality result in the framework of divide-and-conquer algorithms: for such an algorithm, one chooses an index q between 1 and p, one deals first with the sequence $x_q, \ldots, x_p$, and next with the sequence $x_1, \ldots, x_{q-1}$. Grimm, Pottier and Rostaing-Schmidt [3] considered all the algorithms and showed that algorithms of divide-and-conquer type provide the optimal time of computation T. In practice, it is necessary to find a trade-off between r, the number of registers, and T, the number of computations, hence the important quantity is the product rT. Grimm, Pottier and Rostaing-Schmidt gave a lower bound for the product $(r+1)T$, which is rather tight and shows that the product rT has order $p \ln^2 p$.

```
>  >  S                          X_6 = (0,3,5,6)
         >  S                    X_5 = (0,3,5)
               P                 X_4 = (0,2,3,4)
            R                    X_3 = (0,2,3)
         R                       X_2 = (0,2)
P                                X_1 = (0,1)
```

FIGURE 1. The diagram show the reversing of a sequence $x_0, \ldots, x_6$ with 3 registers. Column j corresponds to term x_j for $j = 1, \ldots, 6$. The symbol $>$ means that the value is computed but not stored; the symbol S means that the value is computed and stored; P means the values is calculated, printed and then thrown away; R means the value is retrieved from memory, printed and then thrown away. The list X_j gives the indices k of the values x_k which are stored just before term x_j is printed. The total number of symbols $>$, S or P provides the time of computation.

1. Reduction to divide-and-conquer algorithms

The search for an optimal algorithm needs a careful definition of what is an algorithm in this context. The following definition is proposed.

DEFINITION 1. A reversal table of the sequence $x_0, \ldots, x_p$ with r registers is a family $(X_{i,j})$, $0 \le i \le r_j$, $0 \le j \le p$, such that

- $X_{i,j} < X_{i+1,j}$ for $0 \le i < r_j$, $0 \le j \le p$;
- $X_{0,j} = 0$ and $X_{r_j,j} = j$ for $0 \le j \le p$;
- $r_j \le r$ for $0 \le j \le p$.

The definition must be understood in the following manner. The list $X_j = (X_{i,j})_{0 \le i \le r_j}$ provides the values x_k which are stored just before the value x_j is printed. More precisely the list contains the indices k arranged in increasing order. See Figure 1 for an example. Notice that the value x_0 is stored for free because the register used is not taken into account; in fact there are $r + 1$ registers used.

To each reversal table $X = (X_{i,j})$ is associated its time of computation

$$t_X = \sum_{\substack{0 \le i \le r_j \\ 0 \le j \le p}} t_{i,j}, \qquad \text{with} \qquad t_{i,j} = X_{i,j} - Y_{i,j},$$

where $Y_{i,j}$ is the maximal index of stored values less than $X_{i,j}$. Line 5 of Figure 1 provides the following values: $t_{4,2} = 2$ because the value x_2 may be obtained at this time only from x_0; $t_{4,3} = 0$ because x_3 is available from memory, and $t_{4,4} = 1$ because x_4 must be computed from x_3. The goal is to find a reversal table X which provides the minimal time of computation t_X. The main theorem is stated as follows.

THEOREM 1. *There exists an optimal reversal table which is of divide-and-conquer type.*

We say that a reversal table $(X_{i,j})$ is of divide-and-conquer type if there exists an index q such that

$$X_{1,p} = X_{1,p-1} = \cdots = X_{1,q} = q.$$

This means that the algorithm computes the value x_q, handles the sequence $x_q, x_{q+1}, \ldots, x_p$, and next the sequence $x_0, \ldots, x_{q-1}$.

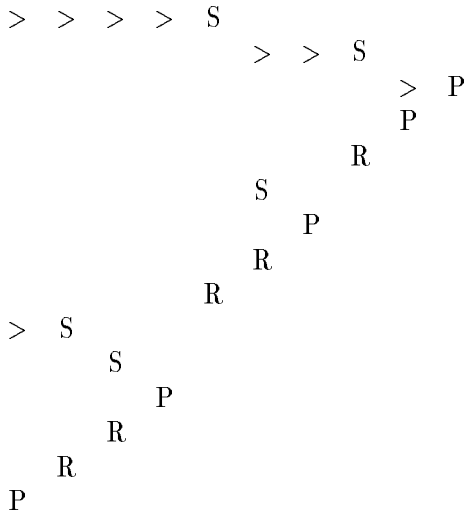


FIGURE 2. The diagram shows a divide-and-conquer optimal reversing of a sequence of length $p = 10$ using $r = 3$ registers. The time of computation is $T = 18$.

2. Optimal time

The previous result reduces the search for an optimal algorithm to the consideration of algorithms of divide-and-conquer type.

THEOREM 2. *The time of any optimal reversal table of a sequence $x_0, \ldots, x_p$ with r registers is given by*

$$T(p, r) = k(p + 1) - \binom{r + k}{r + 1},$$

where k is any integer which satisfies

$$\binom{r + k - 1}{r} - 1 \leq p \leq \binom{r + k}{r} - 1.$$

Moreover a reversal table of divide-and-conquer type is optimal if and only if its index q satisfies

$$\binom{r + k - 2}{r} \leq q \leq \binom{r + k - 1}{r}, \qquad \textit{and} \qquad \binom{r + k - 1}{r - 1} - 1 \leq p - q \leq \binom{r + k - 1}{r - 1} - 1.$$

The first part of the assertion appears in [1]. The proof uses an auxiliary function $m_{r,s}$; this function gives the maximal length of a sequence which can be inverted using only r registers and computing only s times each value x_k in the worst case. The proof of the second part relies on the consideration of

$$f(q) = q + T(q - 1, r) + T(p - q, r - 1).$$

This function of the real variable q achieves its minimum on the interval given in the theorem and this minimum is $T(p, r)$. This gives a functional equation for $T(p, r)$, which translates exactly the divide-and-conquer strategy.

It must be noted that for a divide-and-conquer optimal reversal table the number r of registers is exactly the maximal number of times a term of the sequence is computed. One can observe this phenomenon in the example of Figure 2, where the terms $x_1, \ldots, x_{10}$ are respectively computed 3, 2, 2, 2, 1, 1, 2, 2, 1, 2, 1 times.

3. Space-time trade-off

Up to now the number r of available registers was fixed. But it is natural to make the computation more efficient by choosing r as a function of p. In this context the quantity of interest is the product $rT(p,r)$.

THEOREM 3. *The product $(r+1)T$ is greater than a quantity which is equivalent to $p\ln^2 p\ln^{-2} 4$. There exist arbitrary large p's and r's such that the product $(r+1)T$ is equivalent to $p\ln^2 p\ln^{-2} 4$.*

The idea of the proof is to replace the true quantities using the approximations

$$(r+1)T(p,r) \simeq (p+1)r(k-1), \qquad \binom{r+k}{r} \simeq (r+k)^{r+k} r^{-r} k^{-k}.$$

This gives an r which minimizes the product $(r+1)T$. The result is illustrated by Figure 3.

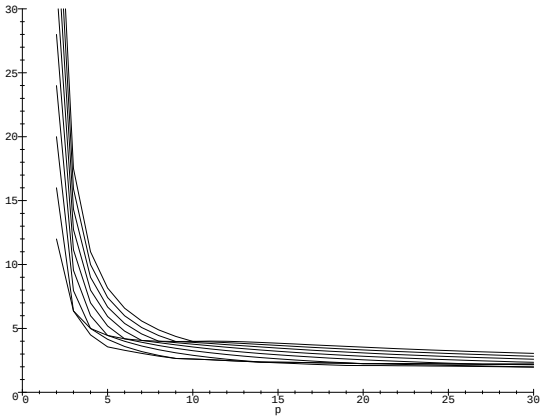


FIGURE 3. The product $rT(p,r)$ is close to $C_p = p\ln^2 p\ln^{-2} 4$ for p large. Shown here are the sequences $rT(p,r)/C_p$ for $1 \leq p \leq 30$ and $r = 2, \ldots, 10$.

Bibliography

[1] Abbott (J.) and Galligo (A.). – Reversing a finite sequence. – Preprint, December 1991.

[2] Baur (W.) and Strassen (V.). – The complexity of partial derivatives. *Theoretical Computer Science*, n° 22, 1983, pp. 317–320.

[3] Grimm (J.), Pottier (L.), and Rostaing-Schmidt (N.). – A sharp lower bound on the time-space product for reversing a finite sequence. – Preprint, 1995.

[4] Morgenstern (J.). – How to compute fast a function and all its derivatives, a variation on the theorem of Baur-Strassen. *SIGACT News*, n° 16, 1985, pp. 60–62.

A Computer Support for Genotyping by Multiplex PCR

Pierre Nicodème
INRIA-Rocquencourt

January 16, 1995

[summary by Pierre Nicodème]

Abstract

The Polymerase Chain Reaction, PCR for short, is able to produce million copies of a specified DNA segment. Grouping (multiplexing) numerous PCR in a few experiments would decrease the PCR costs and save time. Starting from a biological model for the multiplexing conditions, we transform the problem to a combinatorial one, show that the problem is $NP-$complete, give an approximation algorithm, and show its quasi-optimality.

1. Introduction

Devised in the mid-1980s, the Polymerase Chain Reaction, PCR for short[1], is able to produce enormous numbers of copies of a specified DNA sequence. The method is sensitive to very small amounts of DNA, and has numerous applications (diagnostics, etc); however, in most of the PCR experiments performed by biologists, the amplification of each target fragment of DNA requires a separate and costly PCR experiment, with the corresponding manipulations, and the immobilization of an automat [4].

PCR exploits certain features of DNA replication. Single-stranded DNA is used as a template for the synthesis of a complementary new strand. These single stranded DNA templates can be produced by simply heating double-stranded DNA to temperature near boiling. Then we require a small section of double stranded DNA to initiate ("prime") synthesis.

The starting point for DNA synthesis can be specified by supplying an oligonucleotide primer that anneals to the template at this point. Both DNA strands can serve as templates for synthesis provided an oligonucleotide primer is supplied for each strand. Each cycle of PCR duplicates the segments under amplification; so, starting from one segment, n cycles of PCR produce 2^n segments. Figure 1 shows the synthesis initiated by the forward primer 5'-ACACA...AGCAA-3' on the 3'-5' strand of a segment of DNA[2].

Primers cannot be chosen at will inside a locus (a portion) of a gene: they must respect conditions permitting a correct amplification by PCR; the temperature of hybridization at which the polymerase synthesises the new DNA strands is one of these conditions; this temperature depends on the composition of the primer, and more specifically on the respective percentage of the bases A and T, versus the bases G and C; a more accurate method relates the hybridizing temperature

[1] We refer to [5] for a detailed introduction to the subject of PCR.

[2] The 3' extremity of a chain is N-terminal; the 5' extremity is C-Terminal; the numbers 3 and 5 refer to the position of the carbon connected to the N-termination and to the C-termination inside the 5-carbon sugar constitutive of the bases of DNA (other components of a base of DNA are a phosphate group and one out of four organic bases).

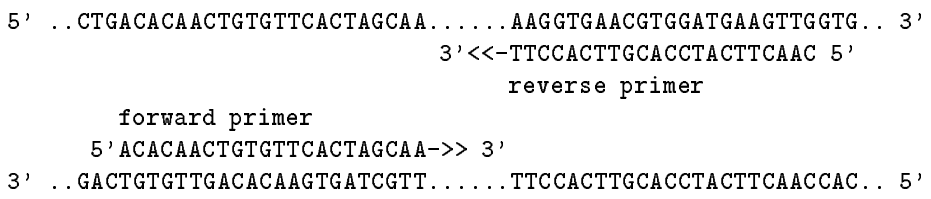


FIGURE 1. Primers for DNA polymerase

to experimental measurements of base-stacking energy. Anyhow, when choosing a pair of primers, the hybridizing temperatures of the two primers should be about the same. Another condition relates to homology between the two primers and to self-homology; such homology would very often prevent a correct amplification, the primers hybridizing to each other, or identical copies of a self-homologous primer hybridizing together.

Several software programs are available to predict which pair of primers to choose inside a given locus. The conditions which hold for a one-locus PCR amplification still have to hold for multi-loci amplification.

Starting from a set S of n loci, we want to find the subset C_{max} of maximum size of S, such that in each locus of C_{max} we can select a pair of compatible primers, and such that the $2n$ selected primers are each other compatible.

We made an extension the program PRIMER, of S. E. Lincoln, M. J. Daly, and E. S. Lander [1] in a MULTIPCR program. PRIMER is a two-step program; step-1 selects forward and reverse candidates primers; step-2 chooses a best pair of one forward and one reverse primer among all the possible pairs of candidates. MULTIPCR takes as input the output of PRIMER step-1, and chooses for each locus a forward and a reverse primer compatible with the primers chosen for the other loci, whenever this is possible.

2. Multiplexing the Polymerase Chain Reaction

2.1. Requirements. We detail in this section a model of compatibility between primers that Gilles Thomas[3] proposed to us and the corresponding requirements.

We will speak of *locus* amplification when considering the amplification of a single segment; only one amplification is allowed inside a given locus; to each locus amplification correspond a *forward* and a *reverse* primer. We define a *subprimer* as a subsequence of a primer and we consider in the following that all subprimers of a multiplexing experiment have the same length σ. In practical experimentations, σ will have values 4 or 5. We define a *3'-subprimer* as the subprimer ending a primer at its 3' extremity (primers being always read in the direction 5' $\Rightarrow$ 3').

The requirements are the following:

(1) *Locus* amplification requirements:
   (a) The distance between the forward primer and the reverse primer is between 150 and 450 bases (these minimum and maximum values are given as parameters and correspond to the "product range size" taken as input by the program PRIMER).
   (b) The primers satisfy the conditions of non-palindromicity; such a palindromicity would cause self-homology.
   (c) The 3'-subprimers are not reverse complementary with any of the subprimers (subprimers as 3'-subprimers are assumed to be of length σ bases).

(2) *Multi-locus* amplification or *experiment* requirements:

[3]Laboratoire de Génétique des Tumeurs, Institut Curie, 26, rue d'Ulm, 75005 Paris.

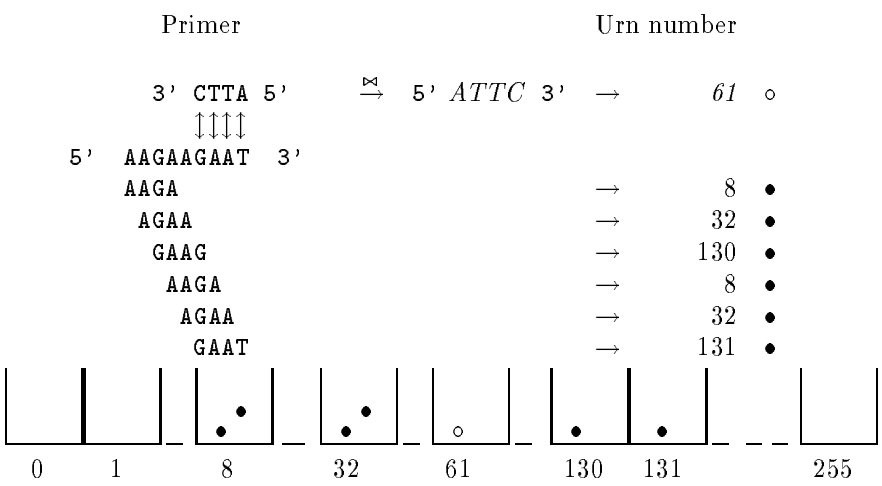


FIGURE 2.

(a) Any 3'-subprimer of an experiment is not reverse complementary with any subprimer, including itself, of any primer of the experiment; this would initiate hybridization of the primers themselves. An example for this condition is given in subsection 2.2. Note that subprimers (including the 3'-subprimers) may be identical between different loci, or inside a locus.

(b) The temperatures of denaturation, or the GC/AT percentage in the primers of a multi-locus PCR amplification have to belong to a limited range of values (by instance $48\% - 52\%$).

(c) Electrophoresis[4] distance: the difference of lengths between any two segments amplified in the same multi-locus PCR amplification is greater than δ bases; this is necessary to allow a correct differentiation of the amplified segments after electrophoresis. This distance supposes that the loci are not polymorphic, in which case the problem of differentiating the amplified segments has to be handled in a different way.

2.2. An urn model to solve the problem of compatibility between primers. We give here a constructive example of our algorithm.

- Using the mapping (A $\Rightarrow$ 0, C $\Rightarrow$ 1, G $\Rightarrow$ 2, T $\Rightarrow$ 3), We transform each subprimer of length $\sigma = 4$ in a number in base 4 between 0 and $4^4 = 256$, and each subprimer of length $\sigma = 5$ in a number between 0 and $4^5 = 1024$; the resulting numbers are converted in base 10 (TTA $\Rightarrow 330_4 \Rightarrow 60_{10}$).
- We then consider a model of 256 urns, when $\sigma = 4$, or a model of 1024 urns, when $\sigma = 5$. For each subprimer, we compute the associated number as described above, and we throw a ball in the corresponding urn.

The compatibility constraint (requirement 2(a) of §2.1) is then transformed as shown in Fig. 2 (when $\sigma = 4$). The complementary of the 3'-subprimer is taken (CTTA in Fig. 2) and reversed

[4]Electrophoresis is a migration method which allows short segments to move faster than the long ones; this method allows the differentiation of segments of different lengths, from a mixture of them, but it has a limited precision corresponding to our parameter δ.

(ATTC in Fig. 2), with $\bowtie$ being the palindromic operation on a chain). Ordinary subprimers generate black balls, while reversed complementary 3'-subprimers generate white balls.

The compatibility rule implies that an urn can never contain simultaneously black and white balls.

2.3. An algorithm deriving from the urn model. We propose in this section an approximate algorithm with high efficiency in practical computations; this algorithm is likely to be almost optimal.

Our algorithm is as follows: we sort our set of loci in increasing order along the number of candidates pairs of primers; we process our set of ordered loci, locus after locus; for each locus, we try each possible pair of primers with respect to the conditions, including the distance condition (requirement 1(a)).

For each pair, we "throw white and black balls in urns", along the model described above; we eliminate the pairs which cause "black and white" collisions; among the acceptable pairs of primers, we select the pair of primers which minimizes, in the following order:

(1) the number of urns containing white balls;
(2) the number of urns containing black balls, whenever the number of "white urns" is identical for two pairs.

The "white and black balls" corresponding to pairs of primers already selected remain in the urns when processing a new locus.

The loci providing no compatible pair with the pairs of the loci already chosen for the current experiment are left apart and processed in a next experiment.

Experimental result shows that, when processing 248 loci of Genbank, it would be theoretically possible to amplify simultaneously 245 loci, with $\sigma = 5$; the average size of the loci is 4000 bp., with an average number of 20,000 admissible pairs of primers. However, it is biologically unrealistic to think to amplify simultaneously much more than ten loci.

3. Determining the pairs of primers which maximize the number of loci in a single experiment is a NP-complete problem

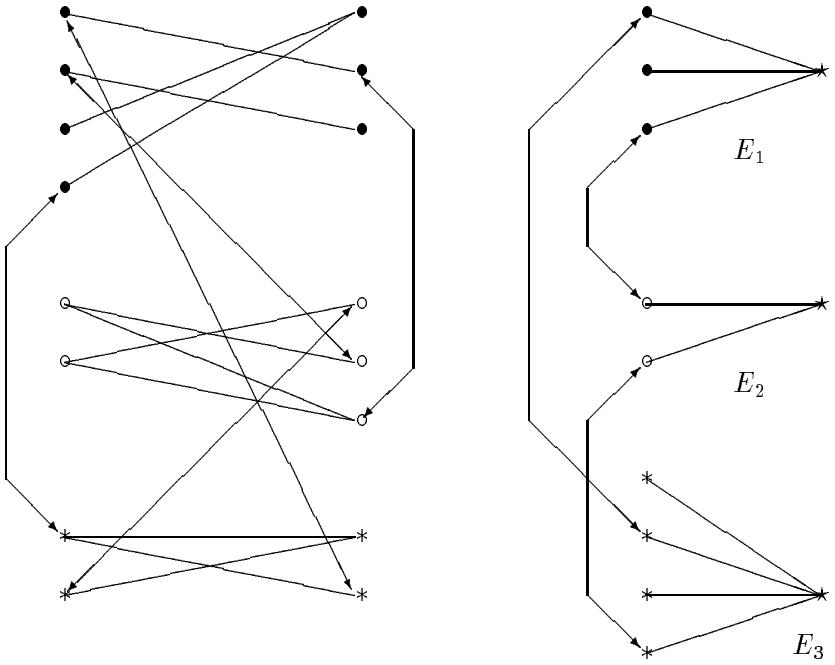
We model our problem as a set of bipartite subgraphs with additional edges (Figure 3 (a)); in this graph, each primer is represented by a vertex; the set of vertices is partitioned by locus, each locus corresponding to a bipartite subgraph; in our example, vertices belonging to the same locus are represented by the same character ($\bullet$ for locus 1, $\circ$ for locus 2, $*$ for locus 3), the forward primers being represented on the left part of the figure (Figure 3 (a)), while the reverse primers are represented on the right part. There are two kind of edges:

- acceptance edges, inside the bipartite subgraph restricted to a single locus; such a non-arrowed edge indicates that the forward and the reverse primers joined by the edge are compatible;
- incompatibility edges, joining a vertex of a locus to a vertex of a different locus; these edges with arrowed extremities indicate that the primers they join are not compatible.

Our "*Compatible Primers Problem*", in short CPP, has the following description:

Instance of the problem: a graph composed of a set of bipartite graphs $B_1, B_2, \ldots, B_J$; the edges of these graphs constitute a set of acceptance edges A; a set of incompatibility edges, these edges joining pairs of vertices which do not belong to the same bipartite subgraphs; an integer K.

Question: is it possible to choose a subset of acceptance edges $A' \subseteq A$ with $|A'| \geq K$ such that A' contains at most one edge from each B_i, $1 \leq i \leq J$, and such that no two vertices belonging to these edges are extremities of an incompatibility edge.



(a) A representation of the compatible primer problem

(b) A transform of the multiple choice matching problem

FIGURE 3. Transformation of any graph to a set of bipartite subgraphs modelling the compatible primers problem

A typical graph of CPP is shown in Figure 3(a); any edge of a *"Multiple Choice Matching Problem"* (in short MCMP) graph is transformed in a vertex of a CPP graph; dummy nodes (figured by $\star$) are added, one for each subset of vertices of MCMP. Figure 3(b) represents such a transform of a MCMP graph (not represented in the figure) to a CPP one. This transformation is detailed in [2, 3]. Therefore, solving CPP in polynomial time would also solve MCMP in polynomial time, which contradicts the NP-completeness of MCMP. We hence proved $NP-$completeness of CPP.

4. Evaluating the limit probability of rejection of a locus

The experimental results obtained with 248 loci show that about 50 loci are enough to fill almost completely the system of urns. We want to evaluate the probability of rejection of a locus in such a saturated system of urns.

if s is the number of subprimers of a primer (practically, if $\sigma = 4$, $s = 17$ for primers of length 20), with $\pi_{1,b,n}$ the probability of acceptation of a primer by a system of U urns containing either white, or black balls, we have

$$\pi_{1,b,n} = \frac{b}{U}\left(1 - \frac{b}{U}\right)^s \quad \text{and} \quad \pi_{1,30,226} = 0.014. \tag{1}$$

The probability π_{11} of compatibility of two primers between themselves, when considering an empty system of urns, is

$$\pi_{11} = \frac{1}{U}\left(1 - \frac{1}{U}\right)^{2s} + \left(1 - \frac{1}{U}\right)\left(1 - \frac{2}{U}\right)^{2s} = 0.766. \tag{2}$$

The MULTIPCR algorithm considers a number V of forward primers for a locus, and, for each forward primer, a number R of reverse primers at an acceptable distance of this primer (between 150 and 450 bp); depending of the locus length, V is between 100 and 500, while R remains close to 50.

The small value of $\pi_{1,30,226}$ allows us to apply the Poisson approximation to the binomial distribution of the number of accepted forward and reverse primers, with respective parameters $\nu = V\pi_{1,b,n}$ and $\rho = R\pi_{11}\pi_{1,b,n}$.

The probability Π of rejection of a locus is then

$$\Pi(\nu(V), \rho(R)) = \sum_{i=0}^{\infty} (\Pr\{v = i\} \times (\Pr\{r = 0\})^i) = e^{-\nu + \nu e^{-\rho}}, \tag{3}$$

probability whose some values for $R = 50$ are

V	250	300	350	400	450
$\Pi(\nu(V), \rho(50))$	0.230	0.171	0.128	0.095	0.071

Considering our experimental results on 248 loci, this shows that our algorithm is quasi-optimal.

Bibliography

[1] Lincoln (S. E.), Daly (M. J.), and Lander (E. S.). – *PRIMER: A Computer Program for Automatically Selecting PCR Primers.* – MIT Center for Genome Research and Whitehead Institute for Biomedical Research, Cambridge, Massachusetts, 1991.

[2] Nicodème (P.). – *A computer support for genotyping by multiplex PCR.* – Technical Report n° LIX/RR/93/09, LIX, École polytechnique, France, 1993.

[3] Nicodème (P.). – Un support informatique pour le multiplexage de la PCR. *Technique et Science Informatique*, 1995. – Numéro special bioinformatique, à paraître.

[4] Olschwang (S.), Delaitre (O.), Melot (T.), Peter (M.), Schmitt (A.), Frelat (G.), and Thomas (G.). – Description and use of a simple laboratory-made automat for *in vitro* DNA amplification. *Methods in Molecular and Cellular Biology*, vol. 1, n° 3, May/June 1989, pp. 121–127.

[5] Watson (J. D.), Witkowski (J.), Gilman (M.), and Zoller (M.). – *Recombinant DNA.* – Scientific American Books, 1992, 2nd edition, 79–98p.

Genomic Sequence Comparison

Pavel Pevzner
Computer Science Department
The Pennsylvannia State University
University Park, PA 16 802

June 26, 1995

[summary by Mireille Régnier]

1. Introduction

Sequence comparison is traditionally based on gene comparison based on local mutations (insertions, deletions or substitutions of nucleotides). Such comparisons do not yield evolutionary information. It appears that evolution is manifested as the divergence in gene order. For example, the large number of conserved segments in the maps of man and mouse suggests that multiple chromosome rearrangements have occurred since the divergence of lineages leading to human and mice. The number of such rearrangements has been recently estimated to be approximately 180. This leads to a major shift of sequence comparison toward the analysis of such rearrangements at the *genome level*. Nevertheless, there are almost no computer science results allowing a biologist to analyze gene rearrangements.

This talk addresses two problems: define and estimate the distance between two different species for a same gene and reconstruct the rearrangement scenario. A paper version can be found in [1].

2. State of the Art

Some genomes evolve so rapidly that the similarity between many genes is very low and is indistinguishable from the background noise. Nevertheless, according to Ohno's law, gene content of X chromosomes is assumed to have remained the same throughout mammalian development in the last 125 million years. However, the order of genes on X chromosomes has been disrupted several times, even though synteny has been almost completely conserved.

The order of genes in two organisms is represented by permutations $\pi = (\pi_1 \pi_2 \dots \pi_n)$ and $\sigma = (\sigma_1 \sigma_2 \dots \sigma_n)$.

DEFINITION 1. A *reversal* ρ of an interval $[i, j]$ is a permutation

$$\rho = [1, 2, \dots, i-1, j, j-1, \dots, i+1, i, j+1, \dots, n]$$

$\pi.\rho$ has the effect of reversing genes $\pi_1, \pi_{i+1}, \dots, \pi_j$.

The *reversal distance problem* is to find a series of reversals $\rho_1, \rho_2, \dots \rho_t$ such that $\pi \cdot \rho_1 \cdot \rho_2 \cdots \rho_t = \sigma$ and t is minimum (Fig. 1a). The number t is called the *reversal distance*. *Sorting by reversals* is the problem of finding reversal distance $d(\pi)$ between π and identity i.

Reversals generate the symmetric group S_n. Given a set of generators of a permutation group, determining the shortest product of generators that equals π is NP-hard [2]. The problem is PSPACE-complete [4]. And in [5] it is conjectured that sorting by reversals is NP-complete even

when the generator set is fixed. Bounds for the related problem of sorting by prefix reversals can be found in [3]. Gollan conjectured that the reversal diameter of S_n, i.e. the maximal reversal distance between two permutations, is $d(n) = n - 1$, a bound achieved for only one permutation. Lower bound and verification for $n < 200$ are presented in [5].

3. Breakpoint graph

The key idea to sort by reversals is the definition of the *breakpoint graph* (see Fig. 1).

Let $\pi = (\pi_1 \dots \pi_n)$ be a permutation of the elements $\{1, \dots, n\}$. Denote $i \sim j$ if $|i - j| = 1$. Extend a permutation $\pi = (\pi_1 \dots \pi_n)$ by adding $\pi_0 = 0$ and $\pi_{n+1} = n + 1$. We call a pair of consecutive elements π_i and π_{i+1}, $0 \leq i \leq n$, of π a *breakpoint* if $\pi_i \not\sim \pi_{i+1}$. The *breakpoint graph* of π is an edge-coloured graph $G(\pi)$ with $n+2$ vertices $\{\pi_0, \pi_1, \dots, \pi_n, \pi_{n+1}\}$. We join vertices π_i and π_j by a *black* edge if $i \sim j$ and by a gray edge if $\pi_i \sim \pi_j$. (See Fig. 1b). Later we also use the notion of breakpoint graph $G(\pi, \gamma)$ for *two* permutations π and γ which is defined as $G(\pi, \gamma) \equiv G(\pi\gamma^{-1})$ described earlier. A *cycle* in an edge-coloured graph G is *alternating* if the colours of every two consecutive edges of this cycle are distinct. In the following, by cycles we mean alternating cycles.

Let $\vec{\pi}$ be a *signed* permutation of $\{1, \dots, n\}$, i.e. a permutation with "+" or "−" sign associated with each element (Fig. 1c). In the signed case, every reversal of fragment $[i, j]$ changes *both* the order and the signs of the elements within that fragment. We are interested in the minimum number of reversals $d(\vec{\pi})$ required to transform a signed permutation $\vec{\pi}$ into the identity signed permutation $(+1 + 2 \cdots + n)$. Define a transformation from a signed permutation $\vec{\pi}$ of order n to an (unsigned) permutation π of $\{1, \dots, 2n\}$ as follows. To model the signs of elements in $\vec{\pi}$ replace the positive elements $+x$ by $2x - 1, 2x$ and negative elements $-x$ by $2x, 2x - 1$ (Fig. 1c). We call the unsigned permutation π, the *image* of the signed permutation $\vec{\pi}$. In the breakpoint graph $G(\pi)$, elements $2x - 1$ and $2x$ are joined by both black and gray edges for $1 \leq x \leq n$. We define the breakpoint graph $G(\vec{\pi})$ of a signed permutation $\vec{\pi}$ as the breakpoint graph $G(\pi)$ with these $2n$ edges excluded. Observe that in $G(\vec{\pi})$ every vertex has degree 2 (Fig. 1c) and therefore the breakpoint graph of a signed permutation is a collection of disjoint cycles. Denote the number of such cycles as $c(\vec{\pi})$. We observe that the identity signed permutation of order n maps to the identity (unsigned) permutation of order $2n$, and the effect of a reversal on $\vec{\pi}$ can be mimicked by a reversal on π thus implying $d(\vec{\pi}) \geq d(\pi)$. In the following, by sorting the image $\pi = \pi_1\pi_2 \dots \pi_{2n}$ of a signed permutation $\vec{\pi} = \vec{\pi}_1\vec{\pi}_2 \dots \vec{\pi}_n$, we mean sorting of π by reversals $\rho(2i + 1, 2j)$ which "cut" *only after even positions* in π. In the rest of this section, π is an image of a signed permutation.

Cycle decompositions play an important role in estimating the reversal distance. Applying a reversal to a permutation may change the number of breakpoints, $b(\pi)$, as well as the number of cycles in a maximum decomposition, $c(\pi)$. The key idea in the algorithm of [1] is to take advantage of this strong correlation. One proves:

THEOREM 1. *For every permutation π and reversal ρ, one has:*

$$\Delta b(\pi, \rho) + \Delta c(\pi, \rho) \leq 1.$$

PROOF. (sketch): every reversal removes/adds at most two breakpoints. One considers all 5 potential values of Δb in a case-by-case fashion. □

This immediately gives a new lower bound for the reversal distance:

THEOREM 2. *For every permutation π, $d(\pi) \geq b(\pi) - c(\pi)$.*

For all biological examples, one has $d(\pi) = b(\pi) - c(\pi)$. Hence, the use of the breakpoint graph reduces the reversal distance problem to maximal cycle decomposition problem. One shows:

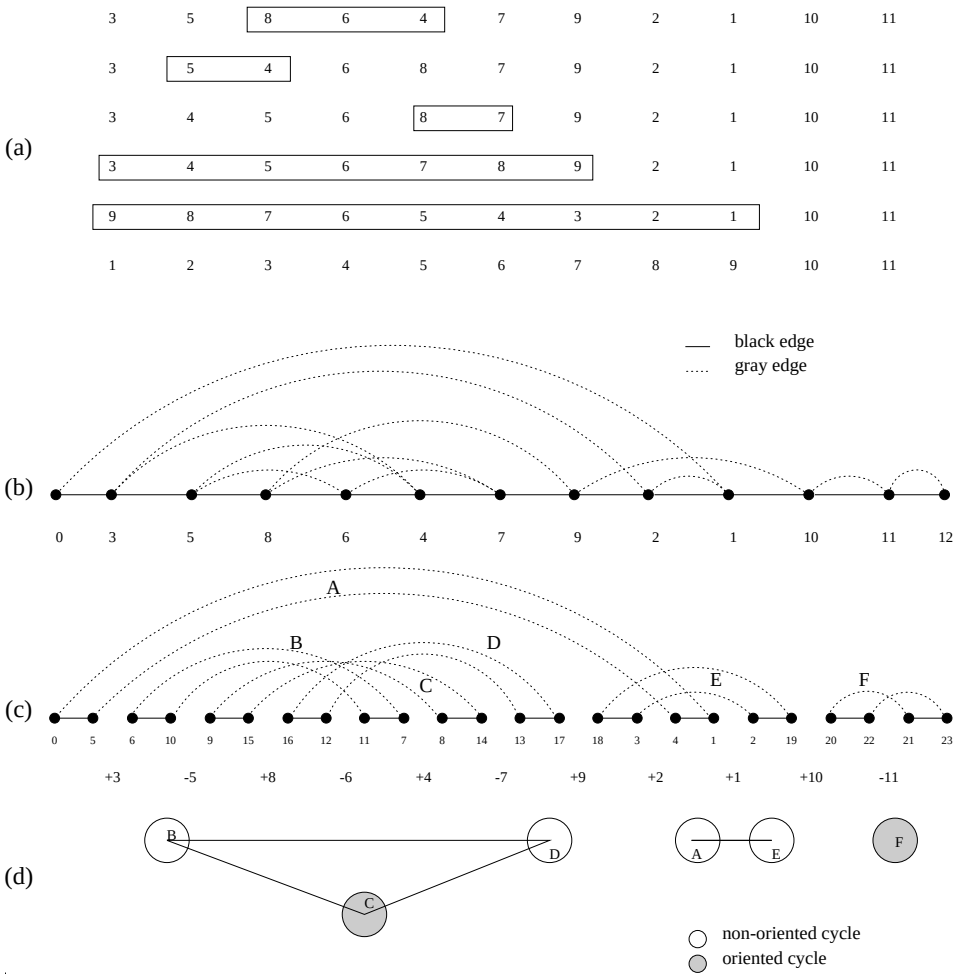


FIGURE 1. (a) Optimal sorting of a permutation $\sigma = (3\ 5\ 8\ 6\ 4\ 7\ 9\ 2\ 1\ 10\ 11)$ by 5 reversals. (b) Breakpoint graph $G(\sigma)$. (c) Transformation of a signed permutation into an unsigned permutation π and the breakpoint graph $G(\pi)$. Gray edges $(8, 9)$ and $(22, 23)$ are oriented while gray edges $(4, 5)$ and $(18, 19)$ are unoriented. Cycles C and F are oriented while cycles A, B, D and E are unoriented. Gray edges $(6, 7)$ and $(12, 13)$ are interleaving while gray edges $(6, 7)$ and $(4, 5)$ are non-interleaving. (d) Interleaving graph H_π with two oriented and one unoriented component.

THEOREM 3 (STRONG GOLLAN CONJECTURE). *For every n, the only permutations that require $n-1$ reversals to be sorted are γ_n and its inverse γ_n^{-1} :*

$$\gamma_n = \begin{cases} (1,3,5,7,\ldots,n-1,n,\ldots,8,6,4,2), & n \text{ even;} \\ (1,3,5,7,n,n-1,\ldots,8,6,4,2), & n \text{ odd.} \end{cases}$$

PROOF. (sketch). Let P_n be the set of n-permutations that satisfy $d(\pi) = n$. It contains γ_n and γ_n^{-1}. One inductively proves that there are no other elements. □

Finally, one proves that the expected reversal distance is very close to the reversal diameter. The key idea is that a typical cycle is long, hence the number of cycles is small. More precisely: $E(d) \geq (1 - \frac{4}{\log n})n$

4. Algorithms

4.1. A greedy algorithm. Define a *strip* of π as an interval $[i,j]$ such that $(i-1,i)$ and $(j,j+1)$ are breakpoints, and no breakpoint lies between them. A strip is *increasing* if π_i, π_j, otherwise it is *decreasing*. A reversal can remove at most two breakpoints; therefore $d(\pi) \geq \frac{b(\pi)}{2}$. In [5] a greedy procedure is given, where one chooses a reversal that removes the most breakpoints of π, resolving ties in favour of reversals that leave a decreasing strip. An upper bound on the number on $d(\pi)$ that provides a performance guarantee of 2, follows from the lemma:

LEMMA 1. *If π is a decreasing permutation with a decreasing strip, then π allows a 1 or 2-reversal. Additionally, If every reversal that removes a breakpoint of π leaves a permutation with no decreasing strips, then π has a 2-reversal.*

4.2. An approximation algorithm for signed permutations. While the problem of sorting signed permutations is easier to handle, it is also more relevant to a biological point of view: genes are directed fragments of DNA sequences. Fortunately, the concept of breakpoint graph as well as strips extends naturally to signed permutations (see above). The algorithm SignedSort sorts a signed permutation π in at most $b(\pi) - \frac{1}{2}c_4(\pi)$ reversals, where c_4 is the number of 4-cycles. It provides an approximation ratio of $\frac{3}{2}$.

4.3. An approximation algorithm for sorting by reversals. As 2-reversals correspond to elimination of 4-cycles, one concentrates on finding a cycle decomposition with a large number of 4-cycles. The algorithm *ReversalSort* achieves an approximation ratio of $\frac{9}{5}$.

To conclude, let us cite among the remaining open problems the analysis of genome rearrangements in *multiple* genomes.

Bibliography

[1] Bafna (V.) and Pevzner (P.). – Sorting by reversals: rearrangements in plant organelles and evolutionary history of mammalian chromosome. *Molecular Biology and Evolution*, vol. 12, 1994, pp. 239–246.

[2] Even (S.) and Goldreich (O.). – The minimum-length generator sequence problem is NP-hard. *Journal of Algorithms*, vol. 2, 1981, pp. 311–313.

[3] Gates (K. W.) and Papadimitriou (Ch.). – Bounds for sorting by prefix reversals. *Discrete Mathematics Algorithms*, vol. 27, 1979, pp. 47–57.

[4] Jerrum (M.). – The complexity of finding minimum-length generator sequences. *Theoretical Computer Science*, vol. 36, 1985, pp. 265–289.

[5] Kececioglu (J.) and Sankoff (D.). – Exact and approximation algorithms for the reversal distance between two permutations. *Algorithmica*, 1995. – To appear.

Part 5

Miscellany

Introduction to Complex Multiplication

François Morain
LIX, École polytechnique

April 10, 1995

[summary by Eithne Murray]

Abstract

The concept of complex multiplication is defined, after some background is given on elliptic functions and quadratic forms. Applications to the class number problem, primality proving and Ramanujan's formulas for $1/\pi$ are presented.

1. Introduction

This is an introduction to the ideas of complex multiplication of lattices and elliptic curves. The theory plays an important role in class field theory, and has had recent applications in algorithmic number theory, especially in elliptic curve primality proving. This presentation is a first glimpse of a very rich and deep theory developed by Kronecker, Weber, Hilbert, Shimura, Deligne, etc [3]. A good introduction to the ideas presented here is [6].

First, some background material on elliptic functions and quadratic numbers is given. This background then allows us to define complex multiplication. Some theorems that demonstrate how these three areas are interconnected and some applications in primality proving, the class number problem and Ramanujan's $1/\pi$ formulas are presented.

2. Elliptic Functions

A *lattice* is an additive subgroup L of $\mathbb{C}$ generated by two complex numbers ω_1 and ω_2 which are linearly independent over $\mathbb{R}$. We write $L = [\omega_1, \omega_2]$. An *elliptic function* for L is a function $f(z)$ meromorphic on $\mathbb{C}$ that is doubly periodic: $f(z + \omega_i) = f(z)$ for $i = 1, 2$.

One of the most important elliptic functions is the Weierstrass $\wp$-function, defined for a lattice L as

$$\wp(z) = \frac{1}{z^2} + \sum_{\omega \in L, \omega \neq 0} \left(\frac{1}{(z-\omega)^2} - \frac{1}{\omega^2} \right).$$

Let G_k, $k \geq 2$, be the *Eisenstein series* for L:

$$G_k(L) = \sum_{\omega \in L, \omega \neq 0} \frac{1}{\omega^k}.$$

Then expanding $\wp(z)$ and $\wp'(z)$ near the origin, we get

$$\wp(z) = 1/z^2 + 3z^2 G_4 + 5z^4 G_6 + \cdots,$$
$$\wp'(z) = -2/z^3 + 6z G_4 + 20 z^3 G_6 + \cdots.$$

Define $g_2 = 60G_4$ and $g_3 = 140G_6$. Then $\wp$ satisfies the differential equation

$$\wp'(z)^2 = 4\wp(z)^3 - g_2\wp(z) - g_3.$$

In addition, $\wp$ and $\wp'$ are generators for the field of elliptic functions over L. One of the main theorems of complex multiplication states when we can write $\wp(\alpha z)$ as a rational function in $\wp(z)$.

In order to easily detect the difference between lattices that are complex multiples of each other and lattices that are truly different, we introduce the concept of j-invariant of a lattice.

For a lattice $L = [\omega_1, \omega_2]$, define $\tau = \omega_2/\omega_1$. Then let $\Delta(\tau) = g_2^3(\tau) - 27g_3^2(\tau)$. The *$j$-invariant* of L is the complex number

$$j(\tau) = 1728g_2^3(\tau)/\Delta(\tau).$$

It turns out that two lattices are isomorphic if and only if they have the same j-invariant.

The j-invariant can also be used to classify elliptic curves [8]. Consider the elliptic curve E, which is an equation of the form $y^2 = 4x^3 - g_2x - g_3$. From the theory of elliptic curves, we know there is a unique lattice L_E such that $g_2 = g_2(L_E)$ and $g_3 = g_3(L_E)$. (The inverse is also true - given a lattice L over $\mathbb{C}$, there is a unique corresponding elliptic curve E_L). We extend the definition of j to elliptic curves by saying that $j(E) = j(L_E)$. Then two elliptic curves E_1, E_2 are isomorphic if and only if $j(E_1) = j(E_2)$.

3. Quadratic Forms

One application of complex multiplication is in solving the class number problem, which is a problem related to quadratic forms. Some background on quadratic forms is needed in order to state this problem. A *quadratic form* is a function $f(x, y) = ax^2 + bxy + cy^2$ with $a, b, c \in \mathbb{Z}$, and its *discriminant* is $-D = b^2 - 4ac$. For future reference, define the *discriminant of a quadratic number* τ to be the discriminant of the unique quadratic form (a, b, c), $a > 0$, $(a, b, c) = 1$ such that τ is the root of $ax^2 + bx + c = 0$.

Define $Q(-D)$ to be the set of all quadratic forms with discriminant $-D$ and $(a, b, c) = 1$. Associate with f a matrix $A = \left(\begin{smallmatrix} a & b/2 \\ b/2 & c \end{smallmatrix}\right)$. Then two quadratic forms f, f' are *equivalent* if there is a matrix $M \in SL_2(\mathbb{Z})$ such that $A' = M^{-1}AM$.

Define $H(-D) = Q(-D)/\sim$, where $\sim$ is the equivalence relation defined above. Then an important theorem states that $h(-D) = |H(-D)|$ is finite. Each class contains exactly one form which is *reduced* ($|b| \le a \le c$, and $b \ge 0$ if $|b| = a$ or $a = c$), and we identify each class with this form: $H(-D) = \{Q_1, Q_2, \ldots, Q_h\}$. Then, there is a (fairly complicated) way of *composing* two forms, and this operation makes $H(-D)$ into an abelian group. $H(-D)$ is called the *class group*, and $h(-D)$ is the *class number*. The problem is to find all $-D$ for which $h(-D)$ is fixed, especially for $h(-D) = 1$. See [7].

4. Complex Multiplication

For two lattices L and M, we say that $L \sim M$ if and only if $\exists \alpha \in \mathbb{C}$ such that $\alpha L = M$. Define $S(L) = \{\alpha \in \mathbb{C}, \alpha L \subset L\}$.

Finally, we can give our definition of complex multiplication.

Definition 1. If $S(L)$ contains more than $\mathbb{Z}$, L is said to have *complex multiplication*. If α is in $S(L) - \mathbb{Z}$, L has *complex multiplication by* α.

$S(L)$ has some special properties.

Theorem 1. *Let $L = [\omega_1, \omega_2]$. Then $\alpha \in S(L), \alpha \in \mathbb{C} - \mathbb{Z}$ if and only if α is a quadratic integer.*

Writing $\tau = \omega_2/\omega_1$, then $\alpha = a\tau + b$, $\tau \notin \mathbb{R}$, so $\mathbb{Q}(\alpha) = \mathbb{Q}(\tau) = \mathbb{Q}(\sqrt{-D})$, where $-D$ is the discriminant of the quadratic number τ. Denote this field by K.

Quadratic field theory tells us that the ring of integers of K is $\mathcal{O}_K = \mathbb{Z}[(-D + \sqrt{-D})/2]$. Then $S(L)$ is a subring of $\mathcal{O}_K$. Thus, if L has complex multiplication by a single element α, then it has complex multiplication by each member of a ring of elements in an imaginary quadratic field. Note that the non-integer elements of $S(L)$ are genuinely complex, which explains the name complex multiplication.

EXAMPLE. Let $Q = (a, b, c) \in H(-D)$, and set $\tau_Q = (-b + \sqrt{-D})/(2a)$. Then the lattice $L_Q = [1, \tau_Q]$ has complex multiplication by all of $\mathcal{O} = \mathbb{Z}[(-D + \sqrt{-D})/2]$.

One remarkable fact is the relationship between lattices with complex multiplication and the $\wp$ function.

THEOREM 2. *Let L be a lattice, and $\wp$ be the $\wp$-function for L. Then L has complex multiplication by $\alpha \in \mathbb{C} - \mathbb{Z}$ if and only if*

$$\wp(\alpha z) = F(\wp(z))/G(\wp(z))$$

with F, G relatively prime polynomials, and $\deg(F) = \deg(G) + 1 = N(\alpha)$.

Algorithms exist to find F and G [6].

Complex multiplication can also be defined on elliptic curves, in the obvious way.

DEFINITION 2. An elliptic curve E has *complex multiplication* if and only if the associated lattice L has.

The following theorem shows which E_Q have complex multiplication, and by what ring.

THEOREM 3. *All elliptic curves $E_Q = \mathbb{C}/L_Q$, where L_Q is defined as in the example, have complex multiplication by the full ring of integers $\mathcal{O}$. These are the only ones with complex multiplication by $\mathcal{O}$, up to isomorphism.*

The following theorem relates complex multiplication with the j-invariant.

THEOREM 4. *Let E be an elliptic curve with complex multiplication, L its associated lattice, and D the associated discriminant. Then $j(E)$ is an algebraic integer of degree $h(-D)$.*

THEOREM 5. *The minimal polynomial of $j(\tau_Q)$, known as the* class equation, *is*

$$H_{-D}(X) = \prod_{Q \in H(-D)} (X - j(\tau_Q))$$

If K is an imaginary quadratic field, then $K_H = K(j(\tau_Q))$ is Galois and is called the Hilbert class field of K.

5. Applications

5.1. ECPP. Elliptic Curve Primality Proving makes use of the class equation to find large prime numbers. Part of the method is to determine if, for a prime p, (p) splits completely in K_H, or equivalently, $H_{-D}(X)$ has h roots mod$p$. Thus one of the problems is to actually compute the class equation. One algorithm to do this involves the theory of complex multiplication [2].

5.2. Class Number Problem. We would like to be able to calculate the number of $-D$'s that have class number h, as well as determine what those $-D$'s are. Work done from 1934 onwards has solved the problem for $h = 1, 2, 3, 4$ and $5 \leq h \leq 23$ for h odd [7, 1].

One method is to consider the minimal polynomial of $j(\tau)$ which has degree $h = h(-D)$. If we can find the minimal polynomial, then its degree will tell us the class number h for the $-D$ associated with τ. The following theorem supplies one approach to the problem.

THEOREM 6. *Let $\gamma_2(z) = \sqrt[3]{j(z)}$ sending $i\mathbb{R}$ to $\mathbb{R}$. If $3 \not| D$ then $\mathbb{Q}(\gamma_2(\tau)) = \mathbb{Q}(j(\tau))$.*

Thus, finding the degree of the minimal polynomial of γ_2 will give the degree of $j(z)$, and hence $h(-D)$. This is an easier task than working with $j(z)$ directly. Various other rather complicated functions (Weber functions [9]) are used in the development of this problem. In particular, the Weber functions allow us to find all imaginary quadratic fields of class number 1.

THEOREM 7. *$h(-D) = 1$ if and only if $d = 3, 4, 7, 8, 11, 19, 43, 67, 163$.*

5.3. Ramanujan. Recalling the Eisenstein series, define E_k by $G_k = 2\zeta(k)E_k$, and then define

$$s_2 = \left(E_2(\tau) - \frac{3}{\pi \Im(\tau)}\right) \frac{E_4(\tau)}{E_6(\tau)}.$$

Then Ramanujan proved

THEOREM 8. *If $\tau \in K$, then $s_2 \in K_H$.*

When combined with an identity from Fricke and Clausen involving s_2, D and $j(\tau)$, some very complicated identities involving π are produced, including the following when $D = 163$:

$$\sum_{n=0}^{\infty} (c_1 + n) \frac{(6n)!}{(3n)!n!^3} \frac{(-1)^n}{640320^{3n}} = \frac{(640320)^{3/2}}{163 \cdot 8 \cdot 27 \cdot 11 \cdot 19 \cdot 127} \frac{1}{\pi}$$

where $c_1 = 13591409/(163 \cdot 2 \cdot 9 \cdot 7 \cdot 11 \cdot 19 \cdot 127)$. This series gives a very fast-converging approximation for $1/\pi$. Other values of D produce similar formulas. See [4] and [5] for more details.

Bibliography

[1] Arno (S.), Robinson (M. L.), and Wheeler (F. S.). – Imaginary quadratic fields with small odd class number. – December 1993. Preprint.

[2] Atkin (A. O. L.) and Morain (François). – Elliptic curves and primality proving. *Mathematics of Computation*, vol. 61, n° 203, July 1993, pp. 29–68.

[3] Borel (A.), Chowla (S.), Herz (C. S.), Iwasawa (K.), and Serre (J.-P.). – *Seminar on complex multiplication.* – Springer, 1966, *Lecture Notes in Mathematics.*

[4] Borwein (Jonathan M.) and Borwein (Peter B.). – More Ramanujan-type series for $1/\pi$. In Andrews (G. E.), Askey (R. A.), Berndt (B. C.), Ramanathan (K. G.), and Rankin (R. A.) (editors), *Ramanujan revisited.* pp. 359–374. – Academic Press, 1988.

[5] Chudnovsky (D. V.) and Chudnovsky (G. V.). – Approximations and complex multiplication according to Ramanujan. In Andrews (G. E.), Askey (R. A.), Berndt (B. C.), Ramanathan (K. G.), and Rankin (R. A.) (editors), *Ramanujan revisited.* pp. 375–472. – Academic Press, 1988.

[6] Cox (David A.). – *Primes of the form $x^2 + ny^2$: Fermat, class field theory, and complex multiplication.* – John Wiley & Sons, New York, 1989.

[7] Goldfeld (Dorian). – Gauss' class number problem for imaginary quadratic fields. *Bulletin of the American Mathematical Society*, vol. 13, n° 1, July 1985, pp. 23–37.

[8] Silverman (J. H.). – *Advanced Topics in the Arithmetic of Elliptic Curves.* – Springer-Verlag, 1994, *Graduate Texts in Mathematics*, vol. 151.

[9] Weber (H.). – *Lehrbuch der Algebra.* – Chelsea Publishing Company, New York, 1902, vol. I, II, III.

Introduction to Simulated Annealing and Boltzmann's Machine

Marcin Skubiszewski
INRIA-Rocquencourt

January 16, 1995

Abstract

Simulated annealing is a technique to find approximate solutions to numerous difficult optimization problems such as NP-complete problems. The method is difficult to use and can be applied to problems whose properties are hardly understood. Boltzmann's machine is a special optimization algorithm designed according to the principles of simulated annealing.

This talk presents simulated annealing with emphasis on the practical use of the method. The presentation of Boltzmann's machine is intertwined with a critical analysis of research on this topic. In particular, we discuss works about practical applications of the algorithm, as well as its use on fast specialized hardware.

A demonstration is given of the working of Boltzmann's machine on specialized hardware. Three optimisation problems illustrate the talk: graph partitioning (the classical computer science MIN CUT problem), synchronizing binary sequences (an open problem from coding theory) and airline traffic planning.

An Algebraic Approach to Residues in Several Variables

Bernard Mourrain
INRIA-Sophia-Antipolis

March 6, 1995

Abstract

We present algebraic methods (mainly from linear algebra) to compute local or global residues of multivariate polynomials. These methods are based on computations of Bezoutians, of which we recall the properties. We show how they make it possible to find the structure of the quotient in the case of a complete intersection and of eigenspaces of the multiplication matrices in this quotient. The relation with "classical" residues will be mentioned.

Contents